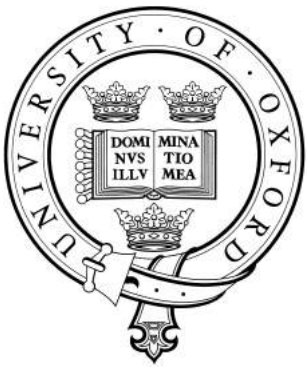
Types for Quantum Computing

Ross Duncan
Merton College, Oxford



Oxford University Computing Laboratory

Submitted for the degree of Doctor of Philosophy

Michaelmas Term 2006

Abstract

This thesis is a study of the construction and representation of typed models of quantum mechanics for use in quantum computation. We introduce logical and graphical syntax for quantum mechanical processes and prove that these formal systems provide sound and complete representations of abstract quantum mechanics. In addition, we demonstrate how these representations may be used to reason about the behaviour of quantum computational processes.

Quantum computation is presently mired in low-level formalisms, mostly derived directly from matrices over Hilbert spaces. These formalisms are an obstacle to the full understanding and exploitation of quantum effects in informatics since they obscure the essential structure of quantum states and processes.

The aim of this work is to introduce higher level tools for quantum mechanics which will be better suited to computation than those presently employed in the field. Inessential details of Hilbert space representations are removed and the informatic structures are presented directly. Entangled states are particularly important in this treatment, as is appropriate, since entanglement is a fundamental driver of quantum computation. The benefits two-fold: as well as producing foundational tools for the study of quantum computation this work also connects quantum mechanics to mainstream areas of computer science such as categorical logic, type theory, program language semantics, and rewriting.

We describe, following Abramsky and Coecke, how quantum mechanics may be carried out without reference to Hilbert space, in a strongly compact closed category. In particular we show how to freely construct a categorical model of abstract quantum mechanics from an arbitrary category.

We introduce *Multiplicative Categorical Quantum Logic* (**mCQL**), a sequent calculus whose proof rules capture the structure of compact closed categories. This sequent calculus is interpreted in a freely generated compact closed category, and its semantics is sound with respect to cut elimination. We define an equivalent graphical syntax, similar to linear logic's proof-nets, and prove that these proof-nets provide a full and faithful representation of any freely generated compact closed category.

Further analysis of the structure of quantum states which correspond to **mCQL** proofs using multiplicative linear logic shows that the linear type system describes the quantum entanglement found in such states. We show that the entanglement present in these states is always of a particularly simple form: collections of entangled pairs.

In order to tackle arbitrary entanglement, we generalise the work of Kelly and Laplaza to give a representation theorem for the free compact closed category by a *polycategory*. Such categories are shown to be equivalent to a generalised system of proof-nets whose axioms may have more than one premise

or conclusion. These axioms may be understood as abstract representatives of interactions involving several distinct quantum systems.

A striking application of entanglement is the class of measurement-based models of quantum computation. In the final chapter, the diagrammatic notation is applied to the verification of programs in the measurement calculus of Danos, Kashefi, and Panangaden — a measurement-based model where the computation is coded directly in an entangled state. By exploiting their diagrammatic form, some example programs are transformed to equivalent quantum circuits, thus proving the correctness of the original programs.

Acknowledgements

It is both an honour and a pleasure to acknowledge my debt to my supervisor, Samson Abramsky. This thesis would not have been possible without his insight, encouragement, patience and generosity.

I also extend my warmest thanks to Bob Coecke, whose friendship and inspiration over the past four years have been invaluable.

Many colleagues have contributed through lectures and discussions from which I have benefited enormously: Prakash Panangaden, Phil Scott, Vincent Danos, Elham Kashefi, Peter Hines, Alexandru Baltag and Eric Paquette, to name only a few. Their ideas have helped shape this work and their interest encouraged me to pursue it. Especial thanks to Ellie d'Hondt for helpful comments on a draft version.

The Oxford University Computing Laboratory provided the necessary financial support for this thesis in the form of a studentship; without this funding I would never have been able to embark on this project. The work was carried out while I was a member of Merton college, which provided generous financial and pastoral assistance. In particular I thank Luke Ong for his support as my tutor and Dean of graduates. I spent six months as a guest of the Ecole Normal Superieure in Paris while writing up; their generosity is acknowledged.

The document was typeset using LaTeX. I made use of Paul Taylor's `diagrams` and `prooftree` macro packages.

In many respects moral support is as important, if not more so, than the financial kind. I owe my sanity to my friends, in Oxford and elsewhere, without whom life would have been very bleak indeed. The value of their laughter, sympathy and kindness cannot be understated.

To my family I express my bottomless gratitude for their love and support throughout my protracted education; and to Nora, for everything, thank you.

Contents

1 Introduction **1**
- 1.1 Quantum Mechanics . . . 5
- 1.2 Programs, Proofs and Categories . . . 10
- 1.3 Outline of the Thesis . . . 12

2 Categorical Background **15**
- 2.1 Monoidal Categories . . . 16
- 2.2 Duality . . . 19
- 2.3 Compact Closed Categories . . . 21
- 2.4 Names and Conames . . . 24
- 2.5 Strong Compact Closure . . . 27
- 2.6 Trace . . . 28
- 2.7 Scalars and Loops . . . 29
- 2.8 Free Construction . . . 31

3 Categorical Quantum Mechanics **35**
- 3.1 Multiplicative Quantum Mechanics . . . 36
- 3.2 **FDHilb** as a strong compact closed category . . . 37
- 3.3 Categorical Quantum Mechanics . . . 39
- 3.4 Free Models . . . 40

4 Multiplicative Categorical Quantum Logic **45**
- 4.1 Formulae . . . 46
- 4.2 Sequent Calculus . . . 46
- 4.3 Proof-nets . . . 55
- 4.4 Example: Entanglement Swapping . . . 65

5 MLL and Entanglement **67**
- 5.1 Entangled States . . . 68
- 5.2 Double Gluing . . . 70
- 5.3 Multiplicative Linear Logic . . . 72
  - 5.3.1 Sequent Calculus . . . 72
  - 5.3.2 Proof-nets . . . 73
  - 5.3.3 Interpreting MLL in the Free Category . . . 74

6 Generalised mCQL **79**
6.1 Polycategories . . . 82
6.2 Graphs and Circuits . . . 87
6.2.1 Graphs . . . 87
6.2.2 Circuits . . . 98
6.3 The Free Compact Closed Category on a Polycategory . . . 104
6.4 Scalars . . . 107
6.5 Homotopy . . . 108
6.5.1 Extended Labellings . . . 109
6.5.2 Homotopy Equivalence . . . 109
6.5.3 Circuits under Homotopy . . . 115
6.5.4 Quotients of the Free Structure . . . 116
6.6 Generalised Proof-nets . . . 117
6.6.1 PN($\mathcal{A}$) is equivalent to $\mathbf{Circ}(\mathcal{A})$ . . . 130
6.7 Relations and Rewriting . . . 133
6.8 Example: Proving No-cloning . . . 135

7 The One-Way Quantum Computer **137**
7.1 The Measurement Calculus . . . 137
7.2 Representing the Measurement Calculus . . . 139
7.3 Examples . . . 146
7.3.1 Teleportation . . . 146
7.3.2 One qubit unitary . . . 146
7.3.3 Controlled-NOT . . . 147
7.3.4 Controlled-U . . . 148
7.4 Remark . . . 148

8 Further Work **151**

Chapter 1

Introduction

The relationship between the quantum computational model and its classical[1] predecessor remains unclear. What are the truly "quantum" features of quantum computing? And how should they be represented? In the following chapters I aim to address these questions by describing the structural features of quantum computation, and other quantum systems, in the terms of logic and type theory.

This thesis is a study of the construction and representation of typed models of quantum mechanics for use in quantum computation. I introduce logical and graphical syntax for quantum mechanical processes and prove that these formal systems provide sound and complete representations of abstract quantum mechanics. In addition, I demonstrate how these representations may be used to reason about the behaviour of quantum computational processes.

The analysis of quantum systems is complicated by the phenomenon known as *quantum entanglement*. Entanglement allows seemingly disjoint systems to behave as a tightly coupled whole, and, as a consequence, quantum systems cannot be understood by simply examining their constituent parts: the entire system must examined together. The exploitation of entanglement is fundamental in quantum computation. It lies at the heart of the speed-up of quantum algorithms, and is used directly in many quantum communication protocols. Indeed, an entire class of quantum computational models, the so-called measurement-based models, is based on the use of entangled resources. Despite all this, the underlying structure of entanglement is poorly understood. A large portion of this thesis is spent developing a mathematical framework which captures the essential features of many-body entanglement in a high level fashion. My aim in so doing is to establish a foundation upon which the behaviour of entangled systems can be understood directly in terms of structural relationships between their subsystems.

A point worth making early is that my focus falls exclusively on the *tensor fragment* of quantum theory, which, despite its fundamental importance, has not been seriously studied before. Other aspects of quantum mechanics, such as non-determinism and branching, are not considered. This sub-theory I refer to as *multiplicative quantum mechanics*, following the terminology of linear logic.

[1] The word *classical* is slightly overloaded when simultaneously discussing logic and quantum computation. Throughout this work I use it to denote *non-quantum*; it will not be necessary to discuss *classical logic* as distinct from linear or intuitionist logic.

The general theoretical backdrop to this work is the Curry-Howard isomorphism, also called propositions-as-types or, more accurately, proofs-as-programs [GLT89, SU06]. This correspondence relates two things: an idealised computing system built up from *program terms* and a *logic* or type discipline to characterise the valid programs. The archetypal example of this correspondence is the connection between the simply typed λ-calculus [Chu40, Bar84, Hin97] and intuitionistic natural deduction [Gen35, Pra65]: every proof defines a λ-term and vice versa. This correspondence is not a static relationship between terms and proofs; it also preserves the dynamical behaviour of program execution. For example, if one λ-term β-reduces to another then their corresponding proofs will be related by the *cut-elimination* procedure for intuitionistic logic. In this fashion logically sound proof-transformations, the most important being cut-elimination, encode program transformations, the most important of which is execution.

The picture is completed by adjoining a third component to this relation: a categorical model which provides *semantics* to formalise the meaning of the programs. In the case of intuitionistic logic, the corresponding model is a Cartesian closed category; the proof-rules of the logic reflect exactly the algebraic structure of the category, hence any model of the simply typed λ-calculus must form such a category [LS86]. The dynamic relationships between terms or proofs given by their normalisation behaviour are reflected in the categorical model as static equalities between arrows. Programs with equivalent behaviour have equal denotations in the model.

The overarching goal of the work carried out in the this thesis is to define a suitable logical syntax, program term language, and categorical model, to construct a similar logical trinity for quantum computation.

The λ-calculus describes only the fundamental actions of abstraction and application, and abstracts away from many details of a practical functional language such as ML [MTHM97] or Haskell [PJH99]. In the same way, we shall avoid getting bogged down in the details of any particular proposed quantum computer by focusing only on their essential common aspect, the theory of quantum mechanics itself.

Quantum mechanics has been studied in logical terms since its earliest days. Traditional approaches follow Birkhoff and von Neuman [BvN36] in taking the lattice of closed subspaces of a Hilbert space as the fundamental object. However this lattice is not distributive and the meet does not have a right adjoint, hence the notion of implication is problematic (see [Sme01], chapter 11 for a discussion). As a result, such quantum logics do not admit a conventional idea of deduction, and thus a Gentzen-style logical calculus is not available. Worse still, it seems impossible to define a tensor product on such lattices, hence the logic is not closed under combination of systems. These difficulties were resolved by Abramsky and Coecke's explicitly categorical account [AC04], which reformulates the axioms of quantum mechanics in the language of strongly compact closed categories with biproducts. Due to its categorical nature, and unlike its predecessors[2], this approach is intrinsically compositional, and hence admits a non-trivial proof theory. Development of this proof theory is a central component of this thesis.

[2]But see also Baltag and Smets's modal quantum logic [BS04] and Isham and Butterfield's topos theoretic approach [IB00].

In order to develop a logical characterisation of quantum processes — or, more accurately, a type-theoretic one — we construct formal, syntactic models of the underlying categorical structures found in quantum mechanics. The axioms of these categories are captured in the syntax exactly, so that the resulting proof-theory may be used to represent and reason about quantum mechanics and, in particular, quantum computation. The central theme is the construction of *free categories* which have the requisite structure. The principal results are:

- We define a sequent calculus, called *Multiplicative Categorical Quantum Logic* (**mCQL**), whose proof rules capture the structure of compact closed categories. Via an equivalent graphical syntax, similar to linear logic's proof-nets, we prove that any freely generated compact closed category has a representation in this logic.

- We use multiplicative linear logic [Gir87a, DR89](**MLL**) to analyse the structure of the quantum states which correspond to **mCQL** proofs and find that their typing in **MLL** describes the quantum entanglement in such states. The entanglement present in these states is always of a particularly simple form: collections of entangled pairs.

- In order to tackle arbitrary entanglement it is necessary to generate the compact closed category on top of an already existing, non-trivial tensor structure. To do so, we step outside of the ground covered by the Kelly-Laplaza coherence theorem [KL80] and consider a larger class of compact closed categories, freely generated upon *polycategories* [Sza75]. We prove a representation theorem and show that these categories are equivalent to a generalised class of proof-nets, whose axioms may have more than one premise or conclusion.

As an example application of this approach, we demonstrate how the generalised proof-net notation can be used to translate between two concrete models of quantum computation. Starting from programs defined the measurement calculus [DKP07], a notation for measurement-based quantum computation, we construct equivalent quantum circuits, and hence verify the correctness of some example programs.

The primary motivation for this work is the need for quantum programming languages. We take for granted that quantum computers will be used to carry out a variety of information processing tasks and that they will form an important class of devices which algorithm designers will call upon. In order to exploit this new paradigm, quantum programmers will require programming languages which allow them to express and manipulate patterns based on the unique features of the quantum model. Since compact closed categories are the fundamental underlying algebraic structure of quantum mechanics it is natural to assume that any type system for a quantum programming language will necessarily represent this structure. Hence we propose that the logical framework introduced here represents a necessary core of any strongly typed quantum language.

Perhaps this claim requires further justification. There have been many proposals for quantum programming languages, c.f. [Kni96, SZ00, Sel04, Öme03, Sab03, AG05], some of which are indeed strongly typed. The most common approach is to take an existing programming language and augment it with an

additional quantum data type, usually qubits; the resulting quantum data are then treated as being essentially unstructured. However, as the quantum teleportation protocol (discussed below) shows, by using entanglement it is possible to construct non-trivial structures in the quantum realm and carry out information processing by manipulating these structures. The logical extreme of this idea is found in the one-way model [RB01], in which the entire computation is carried out by modifying a large entangled state. Hence, if a language is to describe quantum structures — as opposed to qubits embedded in classical programs — then a more serious treatment of such structures is required. With few exceptions [BIP03, DKP07, Per05] this structured approach to entanglement is not found in the existing literature. While this thesis does not contain an exposition of a concrete programming language, it does develop and characterise the necessary algebraic structure for such a language.

The categorical approach yields several other benefits. Since arrows in a category have explicit domains and codomains, everything in the formalism has a *type*, which specifies what kind of thing it is. This contrasts with the approach common in the physics literature where everything, be it a state, unitary transformation, or measurement, is represented as a matrix. Especially useful in a programming environment, but also of more general benefit, is the limitation that types impose on the composition of arrows, which effectively disallows unsound operations. In addition, since the model proposed is abstract and axiomatic, it has interpretations in other settings beyond the usual Hilbert space model of quantum mechanics. In particular, these models — and especially the graphical representations developed here — may be applied in the verification of quantum programs, and used to derive new proof techniques for use in quantum mechanics; see Chapter 7, and also [CP06].

There are also applications of working with a formal logical calculus. When quantum computation was first proposed [Man80, Fey82] it was hoped that quantum computers would be able to efficiently carry out certain tasks which classical machines find difficult or impossible. While there have been some specific examples of practical interest [Sho97, Gro96], the general complexity situation remains unclear [Cle00, AK]. We do not tackle any questions related to algorithms or complexity theory here but, by recasting quantum computation in terms of formal logic, we make a link with the established field of implicit computational complexity. To pick an example near at hand: the syntax we define is closely related to proof-nets for multiplicative linear logic. It has been shown that Boolean circuits of depth n may be simulated by **MLL** proof-nets of depth $O(n)$ [Ter04]. Further, the cut elimination problem for **MLL** is complete for classical polytime [MT03].

The next two sections of this chapter cover some background material necessary for the main body of the thesis. First, in Section 1.1 we describe the basic features of finite dimensional quantum mechanics as formalised in Hilbert spaces. This serves principally to highlight the parts of quantum mechanics of interest, and also to introduce a key example — the teleportation protocol [BBC$^+$93]. After this, Section 1.2 provides a discursive overview of the logical and categorical background to this work. The final section presents an overview of the rest of the thesis.

1.1 Quantum Mechanics

Before immersing ourselves in the abstractions of category theory and categorical quantum mechanics, a very brief review of the postulates of quantum mechanics in their concrete setting is in order. The intent is not to give a full exposition of the theory — see for example [Ish95, Per93, NC00] — but rather to establish the terminology and background assumptions underpinning the following chapters. I should emphasise that I treat quantum mechanics as a *mathematical* theory; in particular I take no position on its interpretation as a description of the physical world, although ideas from quantum information and quantum computation have been influential there. See, for example, [Fuc02, Spe].

Quantum mechanics is the physics of elementary particles — typical objects of consideration are individual atoms, electrons, photons, etc. We will talk simply of quantum mechanical *systems*, defined simply to be any system which obeys the laws of quantum mechanics.

- To each quantum system we associate a finite dimensional complex Hilbert space H, called its *state space*.

In physical problems infinite dimensional state spaces are often encountered; however in the situations typically considered in quantum computation finite dimensions suffice. We will often identify a system with its state space.

- The states of a quantum system are equivalence classes of unit vectors in H; $\psi \sim \phi$ if there exists θ such that $\psi = e^{i\theta}\phi$.

The restriction to unit vectors is required to ensure that the measurement probabilities (see below) sum to one. For other purposes vectors of arbitrary length may be used freely, provided they are normalised before computing the probabilities. With this in mind, we could equivalently define a state to be a *ray* in H containing the vector ψ; that is, a one dimensional subspace

$$\{z\psi \mid z \in \mathbb{C}\}.$$

Working with vectors is more convenient than working with rays, hence we will generally take a concrete representative of each class and defer the normalisation and phase equivalences until the end of any calculation. We will often use the Dirac notation $|\psi\rangle$ to emphasise that ψ is a quantum state; however it will still be treated as a column vector.

- If A and B are the state spaces of two quantum systems, the joint system formed by combining them has state space $A \otimes B$.

The occurrence of a tensor rather than Cartesian product in this postulate makes quantum systems difficult to simulate. Since $\dim(A \otimes B) = \dim A \dim B$, the dimension of the state space of n identical particles increases exponentially in the number of particles. The fact that the dimension of $A \otimes B$ is greater than $A \times B$ essentially amounts to the claim that the space $A \otimes B$ contains elements of the form

$$a_1 \otimes b_1 + a_2 \otimes b_2 \neq a \otimes b$$

which cannot be factorised as a pair for any a or b in A or B respectively. Such states, called *entangled*, represent systems which are correlated with each

other; much of the power of quantum computation derives from entangled states [JL03].

The simplest non-trivial quantum system is the *quantum bit* or *qubit*. Whereas a classical bit is a value from a two element set, a qubit's state space is the 2-dimensional space $\mathbb{C}^2$, which we usually denote Q. The space Q is equipped with a standard basis whose elements are written $|0\rangle, |1\rangle$. The Dirac notation is especially convenient for arrays of qubits: rather than $|0\rangle \otimes |1\rangle$ we write $|01\rangle$. We write $Q \otimes Q$ or $Q^{\otimes 2}$ for the space of two qubits; its standard basis comprises the four vectors $|00\rangle, |01\rangle, |10\rangle$ and $|11\rangle$. Another important basis for this space is the *Bell basis*, which consists of the vectors

$$\beta_1 = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle), \quad \beta_2 = \frac{1}{\sqrt{2}}(|00\rangle - |11\rangle),$$
$$\beta_3 = \frac{1}{\sqrt{2}}(|01\rangle + |10\rangle), \quad \beta_4 = \frac{1}{\sqrt{2}}(|01\rangle - |10\rangle).$$

These vectors are often called *Bell states* or *EPR states*; in particular β_1 is *the* Bell state. The Bell basis is notable because all of its elements are entangled states; due to this property it is very important in many quantum protocols.

Quantum systems change their state via unitary dynamics. More precisely:

- For each discrete time step, the evolution of a closed quantum system is described by a *unitary* map $U : H \to H$.

As I have stated this postulate, it is a consequence of a more general axiom, that quantum systems evolve according to the time dependent Schrödinger equation. However we will only consider discrete time steps here, and simplify accordingly. The unitary evolution maps are treated as a set of basic operations: quantum "gates" for qubits, in analogue with boolean logic gates of conventional electronic circuits.

Since unitaries are isomorphisms, this postulate implies that quantum evolution is reversible. Another important consequence is the following theorem.

Theorem (No-Cloning)**.** *An unknown quantum state cannot be duplicated.*

Proof. Let U be a candidate unitary cloning map; suppose that for some state $|\psi\rangle$ we have

$$U|0\psi\rangle = |\psi\psi\rangle,$$

where $|0\rangle$ is a specially chosen input state. Suppose that U is also able to clone another state ϕ, so

$$U|0\phi\rangle = |\phi\phi\rangle.$$

Since $\langle 0 \mid 0\rangle = 1$, and unitary maps preserve the inner product, we have

$$\langle \phi \mid \psi\rangle = \langle 0\phi \mid 0\psi\rangle = \langle 0\phi| U^\dagger U |0\psi\rangle = \langle \phi\phi \mid \psi\psi\rangle = \langle \phi \mid \psi\rangle\langle \phi \mid \psi\rangle.$$

Hence $\langle \phi \mid \psi\rangle = \langle \phi \mid \psi\rangle^2$, which implies that $\langle \phi \mid \psi\rangle = 0$ or $\langle \phi \mid \psi\rangle = 1$. Hence unitary cloning of quantum states is only possible for states which are elements of a known orthonormal basis. □

Remark. This proof is essentially that found in [NC00]. I include it for comparison with the the abstract version of the no-cloning theorem found in Chapter 6.

A related result, the *no-deleting theorem* [PB00], states that no quantum operation can erase an unknown quantum state.

A closed quantum system is of limited interest as an information processing device. In order to extract information from such a system a *measurement* must be performed. Unlike classical systems, the result of a quantum measurement is not a fixed value determined by the system's state. Instead the state fixes a probability distribution which governs the measurement outcomes. An even more striking difference is that quantum measurements change the state of the measured system.

- A quantum *observable* on a state space H is defined by a self-adjoint linear operator $O : H \longrightarrow H$; in finite dimensions it has spectral decomposition

$$O = \sum_i \lambda_i P_i$$

  where λ_i are the eigenvalues and the P_i are projectors onto the corresponding eigenspace. If a system in state $|\psi\rangle$ is tested against the observable O, the ith outcome will be observed with probability

$$p_i = \langle\psi| P_i |\psi\rangle$$

  and its new state after the measurement will be $\frac{1}{\sqrt{p_i}} P_i |\psi\rangle$.

Since the eigenvalue λ_i occurs neither in the expression for the probability nor in that for the new state, we will ignore it in favour of the index i, which we view as labelling the different possible outcomes of the measurement. It is worth emphasising that a measurement is a state changing operation: if outcome i is observed the transformation

$$|\psi\rangle \longrightarrow \sqrt{p_i} P_i |\psi\rangle .$$

has been performed. This transformation is called the *action* of the measurement. Measurement actions offer an alternative mechanism for the manipulation of quantum systems, particularly entangled systems — this idea is taken to its logical extreme in the one-way model of quantum computation [RB01, RBB02, RBB03].

If the eigenspaces of O are all one dimensional then the choice of a unit vector from each one will determine an orthonormal basis for the space H. Hence, after measuring O, it is guaranteed that the system is in a state corresponding to an element of that basis. Conversely, we will assume that there is a measurement corresponding to any basis for H and speak of, for example, a Bell basis measurement.

A crucial aspect often glossed over in conventional accounts of quantum mechanics is that, after performing the measurement, the experimenter *knows which outcome occurred*. In a traditional physics setting, the eigenvalue associated with the outcome is the actual physical quantity — spin, momentum, or polarisation for example — being measured. In quantum computation we are more often interested in measurement as a part of a larger process and hence a measurement with n possible outcomes produces $\log_2 n$ classical bits of information in addition to the probabilistic state change described above.

The set of outcomes of a measurement is much smaller than the set of states: a continuum is reduced to a finite set. Hence it is impossible[3] to accurately identify an unknown quantum state by measuring it, unless additional information — such as knowing that the state is an element of a certain basis — is provided.

Before moving on, we consider a class of protocols which will occur again and again in later chapters: *quantum teleportation*. .

Typically, a classical bit is transmitted by copying it from the sender's memory to some medium, and then copying it back from the medium into the receiver's memory. The no-cloning theorem prevents quantum bits from being transmitted in a similar way. The limitations of quantum measurement prevent a classical description of the qubit from being transported in its place. Is there any way to transmit qubits other than transporting the physical system encoding them? The teleportation protocol says yes, and shows how entanglement acts as a fundamental substrate in quantum computation.

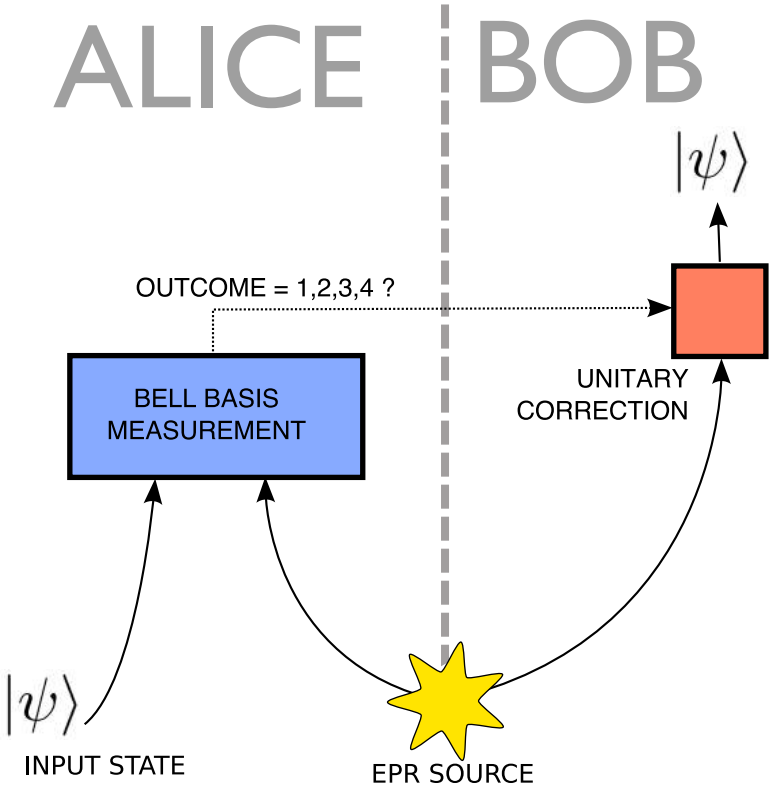


Figure 1.1: The Quantum Teleportation Protocol

The set up is simple and shown in Figure 1.1. Alice has an unknown qubit $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ that she wishes to transmit to Bob. We assume that there is a classical channel between them. As discussed above this will not suffice to transfer the unknown qubit, so in addition we assume that Alice and Bob share a Bell state $|00\rangle + |11\rangle$, with each party possessing 1 qubit. The initial state of

[3]Given an unlimited number of copies of the unknown state, accurate identification of the state is possible — see [Per93] Chapter 3.5.

the combined system is thus:

$$(\alpha\,|0\rangle + \beta\,|1\rangle) \otimes (|00\rangle + |11\rangle) = \alpha\,|000\rangle + \alpha\,|011\rangle + \beta\,|100\rangle + \beta\,|111\rangle$$

Alice performs a Bell-basis measurement upon the two qubits in her possession; there are four possible outcomes depending on which of the four Bell basis vectors she observed. A little calculation shows that if the outcome is the Bell-state $|00\rangle + |11\rangle$ then the new unnormalised state of the system is

$$\alpha\,|000\rangle + \beta\,|001\rangle + \alpha\,|110\rangle + \beta\,|111\rangle = (|00\rangle + |11\rangle) \otimes (\alpha\,|0\rangle + \beta\,|1\rangle).$$

Hence Bob's qubit is now in the original state $|\psi\rangle$.

However, since the measurement action is probabilistic, Alice may observe any of the other three elements of the Bell basis and, indeed, they are all equally likely. The three corresponding states for the system are:

$$\begin{aligned}
\alpha\,|000\rangle - \beta\,|001\rangle - \alpha\,|110\rangle + \beta\,|111\rangle &= (|00\rangle - |11\rangle) \otimes (\alpha\,|0\rangle - \beta\,|1\rangle)\,, \\
\alpha\,|011\rangle + \beta\,|010\rangle + \alpha\,|101\rangle + \beta\,|100\rangle &= (|01\rangle + |10\rangle) \otimes (\alpha\,|1\rangle + \beta\,|0\rangle)\,, \\
\alpha\,|011\rangle - \beta\,|010\rangle - \alpha\,|101\rangle + \beta\,|100\rangle &= (|01\rangle - |10\rangle) \otimes (\alpha\,|1\rangle - \beta\,|0\rangle)\,.
\end{aligned}$$

In each case Bob's qubit is related to the original qubit by a unitary map. Let

$$U_2 = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}\,, \qquad U_3 = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}\,, \qquad U_4 = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}\,.$$

If Alice observes the outcome β_i then Bob can recover the original state $|\psi\rangle$ by applying the transformation U_i to his qubit. (Take $U_1 = 1_Q$). If we index the entries in these unitary matrices, u_{00} the top left, u_{01} for the top right, etc, it can be seen that the Bell basis vectors can be expressed as

$$\beta_k = \sum_{i,j \in \{0,1\}} u_{ij}^{(k)}\,|ij\rangle\,.$$

In other words, the entangled state which is observed by Alice is a representation of the linear map which has been applied between the input and the output.

This can be generalised. Notice that the entangled state initially shared by the participants in the teleportation protocol is β_1, which represents the identity matrix. If they had shared an arbitrary entangled state, say

$$|A\rangle = a_{00}\,|00\rangle + a_{01}\,|01\rangle + a_{10}\,|10\rangle + a_{11}\,|11\rangle\,,$$

then performing the protocol with this resource will result in Bob's qubit finishing in the state

$$|\psi_{\text{Bob}}\rangle = AU_i\,|\psi\rangle$$

where $A = \begin{pmatrix} a_{00} & a_{01} \\ a_{10} & a_{11} \end{pmatrix}$. Hence entangled states can be used to perform arbitrary linear transformations between states.

The connection between entangled states and linear maps is due to the isomorphism between the tensor product of two vector spaces and the space of linear maps between them. This fundamental algebraic fact is central to our approach.

1.2 Programs, Proofs and Categories

The no-cloning and no-deleting theorems put striking limitations on the abilities of a quantum device when compared to the freedom that a classical computer enjoys to duplicate and erase information. What consequences does this have for a theory of quantum types?

In logic the ideas of copying and deleting are expressed through the sequent rules *weakening* and *contraction*.

$$\frac{\Gamma \vdash \Delta}{A, \Gamma \vdash \Delta}\ \text{Weak.} \qquad \frac{A, A, \Gamma \vdash \Delta}{A, \Gamma \vdash \Delta}\ \text{Cont.}$$

Contraction corresponds to the idea that, if a fact A is known, we can make use of that fact more than once in a proof — that knowing A once is the same as knowing it twice. From a computational point of view, a program which requires two inputs of type A could equally well accept one input and use it twice. The corresponding λ-term is $\lambda fx.fxx$. On the other hand, weakening allows a proof which does not require the proposition A to demand it; it allows dummy inputs to a program, as in the λ-term $\lambda fx.f$.

While these inference rules are obviously valid in the context of a logical[4] proof, they ignore any notion of cost. If A represents a resource of some kind — whether it be a block of memory to construct a data structure or money for cigarettes — then these rules do not hold. If contraction is forbidden then each input is effectively consumed when it is used and hence may be used at most once. If weakening is rejected then each input must be used: we cannot "keep the change".

Rejection of both weakening and contraction is the starting point for *linear logic* [Gir87a]. While resource sensitive logics had been considered earlier [Lam69], linear logic was the first attempt to resynthesise the traditional intuitionistic logic in these terms. It permits very fine grained distinctions between programs when compared with the usual λ-calculus and, as a result, has found wide application in computer science [O'H91, Abr93, BS94, MOTW95, Hof99].

The most striking feature of linear logic is the splitting of the traditional connectives $\wedge$ ("and") and $\vee$ ("or") into two pairs. The first set, the *multiplicative* conjunction $\otimes$ ("tensor") and disjunction $\parr$ ("par"), have the strict resource preservation properties discussed above. The tensor $A \otimes B$ represents two logically distinct non-overlapping terms. The disjunction $A \parr B$ is more subtle; in the same way as classical logic defines $A \Rightarrow B \equiv \neg A \vee B$, the linear implication is defined as $A \multimap B \equiv A^{\perp} \parr B$ (where $A^{\perp}$ is the linear negation). Hence $A \parr B$ can be understood as a map from $A^{\perp}$ to B which uses its argument exactly once. Of course $\parr$ is symmetric so it can equally be viewed as a map from $B^{\perp}$ to A, so we see a term of type $A \parr B$ as a channel or correlation between A and B, which may be written to by either side.

The other operations of linear logic, the *additive* connectives & ("with") and $\oplus$ ("plus"), and the *exponential* modalities ! and ? will not concern us here, but the subsystem containing only the multiplicative connectives — called multiplicative linear logic and abbreviated **MLL** — will be of particular relevance.

[4]Note that $\lambda fx.f$ is the K combinator of combinatory logic; when combined with $S := \lambda xyz.xz(yz)$ every computable function can be represented [CF58]. Note also that the input z is used twice in the S combinator.

Perhaps the most fundamental rule found in logic is the *Cut* rule:

$$\frac{\Gamma \vdash A \qquad\qquad A, \Delta \vdash B}{\Gamma, \Delta \vdash B}\ \text{Cut}$$

The cut allows proofs of intermediate results to be combined in the proof of a larger theorem. Viewing proofs as programs, it allows the construction of large programs by composing smaller ones — the output of one process is used as input to another. Linear logic enjoys the *cut elimination* property: its proofs can be transformed so that the conclusions proceed directly from the axioms without any use of cut. We view this process as the execution of the program[5]. However, as is usually the case for sequent calculi, there is an undesirable element of choice involved in the transformation process. The sequential nature of a proof forces a definite order on the inferences when there is no logical reason that one should precede another. The result is that a proof may have many cut-free forms, and many proofs which ought to be equivalent are syntactically distinct.

The multiplicative fragment of linear logic resolves this problem via a beautiful theory of *proof-nets* [Gir87a, DR89, BCST96]. A proof-net is akin to a λ-term; it represents the the logical dependencies of a proof—and nothing more—in the form of a graph. Every **MLL** proof can be translated to a proof-net and vice versa, although the process is not an isomorphism since proofs which have the same logical operations in a different sequence will produce the same proof-net. Proof-nets enjoy strongly normalising cut-elimination, so every net can be reduced to a unique cut-free form, unlike the sequent calculus. Since the normal forms are unique, and the normalisation procedure can be performed very efficiently, the question of equivalence of proofs can be decided simply by normalising and checking if the normal forms are equal. The question of proof-nets for **MALL** is much more complicated [Gir96, LTdF04] but it has recently been resolved [HvG03].

We turn our attention from syntax to semantics. For each logic we can construct a category where the propositions are objects and the proofs are arrows [Lam68, Lam69]. If an arrow exists between two objects A and B then there must be a proof of B based on the assumption of A. In some cases, for example classical logic, this is all that can be said: the structure of the model encodes provability and nothing more. Intuitionistic logic and linear logic, on the other hand, enjoy models where distinct arrows correspond to distinct proofs. If two proofs are equivalent, in the sense of having the same logical content or, more formally, the same normal form, then they are represented by the same arrow. The category is then said to be an interpretation, or model, of the logic. The model will not usually be unique but the structure of the proofs will force every model to have certain algebraic structure. In the case of multiplicative linear logic the requisite structure is a $*$-*autonomous category* [Bar79, See89, Bar91].

Examples of $*$-autonomous categories are common throughout mathematics. For our purposes, the most important is **FDHilb**, the category of finite dimensional Hilbert spaces and linear maps, since concrete quantum mechanics is carried out here. **FDHilb** has a self-dual tensor product, which means that the multiplicative connectives tensor and par have the same interpretation. Hence the maps from an object A to another object B are representable as elements of the tensor product $A \otimes B$ — just as was observed above for entangled

[5] See [Tak87, GLT89, Tro91] for discussion of cut elimination and related proof theory.

states and linear maps. Categories with this property are called *compact closed* [KL80].

In their seminal paper [AC04] Abramsky and Coecke point out that the defining axiom of the compact closed structure implies that the teleportation protocol can be carried out in any such setting. More generally, since every arrow has a representative in a tensor object, there is an analogue to entangled states in all such categories. In the above cited paper, the authors carry out a complete axiomatisation of finite dimensional quantum mechanics in the framework of compact closed categories with biproducts. The quantum part of the system is represented by the degenerate multiplicative framework of the compact closed structure, while the non-determinism and classical choice is represented by the biproduct, which corresponds to a degenerate version of linear logic's additive connectives and has the properties of both a product and a coproduct. This categorical presentation has been refined by later work [AC05, Abr05, Coe05, Sel05, CP07] but the compact closed structure retains a fundamental role, particularly for the analysis of entangled states.

Until now, little work has been done on the logical presentation of such categories aside from Kelly and Laplaza's definitive paper [KL80], and the unpublished [Shi96]; a more recent contribution is [Abr05]. More attention has been directed toward techniques for "repairing" a compact closed category by adding extra structure, in order to produce a non-degenerate model of linear logic [Loa94, AGN95, Tan97, HS03]. In computer science, compact closed structures have been studied in the context of typed concurrency theory as *interaction categories* [AGN96].

The next chapter will describe compact closed categories and their basic properties in detail.

1.3 Outline of the Thesis

The thesis is organised as follows.

Chapter 2 gathers together the requisite material on the various flavours of monoidal categories which will be required in the subsequent chapters, in particular the basic properties of strongly compact closed categories. Most, though not all, of this material is commonly known, but it is presented here to fix notation and terminology.

In Chapter 3 we revisit the axioms of quantum mechanics, and show that the category **FDHilb** is indeed strongly compact closed. Probabilistic branching is not discussed — we focus exclusively on what might be termed the multiplicative aspect of the theory. Within this setting we explain how Abramsky and Coecke's categorical formulation maps the components of the physical theory onto the algebraic structure of strongly compact closed categories and reframe the postulates of quantum mechanics in this language. Finally we consider how models of these axioms with desired equational properties may be constructed, and how the models so constructed are related to the standard interpretation of quantum mechanics in Hilbert spaces.

With the necessary background covered, Chapter 4 begins our main subject: the logic of compact closed categories. However it is not possible to talk about *the* logic of compact closed categories, in the way that intuitionistic logic is *the* logic of Cartesian closed categories. Compact closed categories were first

seriously studied during Kelly's investigation of coherence theorems [Kel72b, Kel72a, Kel74]. They are unusual in that the equations which hold in a compact closed category cannot be expressed purely in terms of functors and generalised natural transformations, in the way that, for example, symmetric monoidal closed categories can [ML63, KML71]. As discussed in the remarks at the end of [KL80], and in [Kel92], compact closed categories do not arise from a "club", and hence the best form of coherence available for them is a description of the free compact closed category generated by another category. The upshot of this for a logical presentation of such categories is that any proof system must contain some non-logical axioms to represent the generators of the free structure.

In Chapter 4, we define a sequent calculus for *Multiplicative Categorical Quantum Logic* (**mCQL**), a formal logic whose inference rules encode the algebraic structure of a compact closed category and whose non-logical axioms are drawn from an arbitrary category $\mathcal{A}$. We define an interpretation map from **mCQL** proofs into the free compact closed category generated by $\mathcal{A}$, and prove that this semantic is sound with respect to cut elimination. We then define a proof-net calculus for **mCQL** and prove its equivalence to the sequent presentation. Finally we demonstrate that every arrow in the free compact closed category generated by $\mathcal{A}$ has a representation as an **mCQL** proof-net. Hence the syntax of **mCQL** is a sound and complete representation of the free category.

One might hope that a theory of quantum types would be able to distinguish between those states which are entangled and those which are not. Unfortunately the entangled states in a given state space do not form a closed subspace, so there is no hope that the objects of the **FDHilb** will provide the desired type information. We must abandon the idea that the entanglement will be abstractly captured by the objects of a compact closed category. However, the syntax of **mCQL**, both in its sequent and proof-net formulations, is a relaxation of the rules of **MLL**, essentially by identifying the connectives of tensor and par. In Chapter 5 we restore that distinction in order to investigate the structure of the entanglement found in these states. We show that, under the duality between points and maps, maximally entangled states correspond to unitary maps. We use the technique of *double gluing*, introduced by Loader [Loa94] and later developed by Tan [Tan97], and Hyland and Schalk [HS03], to build a *-autonomous category on top of any abstract quantum theory; in this setting the connectives of **MLL** serve to distinguish separable and entangled states: points of a tensor product are never entangled, while points of the par may be. This result is strengthened; in the case where an **mCQL** proof-net can be typed in **MLL**, then the corresponding quantum state is separable if and only if its type is tensor product.

Our analysis of the entanglement found in the quantum states which are representable in **mCQL** exposes a weakness in the theory. By choosing to parameterise the construction of the compact closed structure by a *category*, the only primitive entanglement which can be represented is bipartite, since the entangled states are essentially collections of arrows from the generating category, and an arrow has only two ends. In Chapter 6 we attack this problem by generalising the construction of the free compact closed category relative to a *polycategory*. Unlike the case in Chapter 4 we cannot fall back upon the coherence theorem of Kelly and Laplaza [KL80]; hence we spend a substantial portion of the thesis identifying the structure of the resulting category. We prove a representation theorem, which states that the free compact closed category

generated by a polycategory is equivalent to a certain category of diagrams called circuits.

We then define a proof-net calculus similar to that of **mCQL**, but generalised to accommodate multi-ary axiom links. Unlike the case for **mCQL**, these proof-nets do not enjoy cut elimination; however it is possible to prove strong normalisation, and that the system of proof-nets whose axioms are drawn from a given polycategory is equivalent to the category of circuits generated by the same polycategory.

The resulting system is able to represent arbitrary entanglement structures and, indeed, over-approximates the entanglement relation. However this can be remedied by taking a quotient with respect to suitable equations. The last technical chapter demonstrates the use of these representations, with the quotient given as explicit rewrites, to prove the correctness of programs in the measurement calculus.

We conclude with some suggestions for future research based on this work.

Chapter 2

Categorical Background

The abstract formulation of quantum mechanics introduced in [AC04] takes place against the background of a *strongly compact closed category*. This structure is the central mathematical object of this thesis. This chapter gathers together the requisite pieces of category theory needed to develop our main results.

Compact closed categories [KL80] are abundant throughout mathematics and computer science. Examples include **Rel**, the category of sets and relations, finitely-generated projective modules over a commutative ring and Conway games (as categorified in [Joy77]). Of course, **FDHilb**, the category of finite dimensional Hilbert spaces is compact closed, indeed strongly so.

Compact closed categories are defined by the presence of *dual*[1] or *adjoint* objects for each object. In any symmetric monoidal category the full sub-category determined by those objects which do have duals is compact closed, as is the sub-category of **Hilb** determined by the Hilbert-Schmitt maps or, more generally, the nuclear maps of tensored *-category [ABP99].

In computer science compact closed categories have been studied in the context of typed concurrency as interaction categories [AGN95, AGN96]; in logic, compact closed categories are degenerate models of multiplicative linear logic [AJ94, Loa94, HS03]; in physics the category of n-dimensional cobordisms, used in topological quantum field, theory is compact closed [BD95]. More examples are easy to find.

Much of the material of this chapter is standard. Mac Lane [ML97] covers monoidal categories and their coherence; Kelly-Laplaza [KL80] provides the basic properties of compact closed categories and the essential coherence theorem. The notion of *strong* compact closure was introduced in [AC04] and refined in [AC05]. Other sources are cited as needed.

[1] We prefer the term *dual* here, to avoid confusion with linear algebraic adjoint which forms the "strong" part of the strongly compact closed categories.

2.1 Monoidal Categories

Definition 2.1. A category $\mathcal{C}$ is *monoidal* if equipped with a functor $\otimes : \mathcal{C}\times\mathcal{C} \to \mathcal{C}$, a distinguished neutral object I, and natural isomorphisms

$$\alpha_{A,B,C} : A \otimes (B \otimes C) \xrightarrow{\cong} (A \otimes B) \otimes C,$$
$$\lambda_A : I \otimes A \xrightarrow{\cong} A, \qquad \rho_A : A \otimes I \xrightarrow{\cong} A.$$

For the associativity morphism α we require that the pentagon

$$\begin{array}{ccccc} A\otimes(B\otimes(C\otimes D)) & \xrightarrow{\alpha} & (A\otimes B)\otimes(C\otimes D) & \xrightarrow{\alpha} & ((A\otimes B)\otimes C)\otimes D \\ \downarrow{\scriptstyle 1\otimes\alpha} & & & & \uparrow{\scriptstyle \alpha\otimes 1} \\ A\otimes((B\otimes C)\otimes D) & & \xrightarrow[\alpha]{} & & (A\otimes(B\otimes C))\otimes D \end{array}$$

commutes. The isomorphisms λ and ρ express the neutrality of I; we require that the following digram commutes:

$$\begin{array}{ccc} A\otimes(I\otimes B) & \xrightarrow{\alpha} & (A\otimes I)\otimes B \\ & {\scriptstyle 1\otimes\lambda}\searrow \quad \swarrow{\scriptstyle \rho\otimes 1} & \\ & A\otimes B. & \end{array}$$

A monoidal category is called *strict* if the isomorphisms α, λ,and ρ are all identities.

Proposition 2.2. *In a monoidal category the equality*

$$\lambda_I = \rho_I$$

holds and the following diagrams commute:

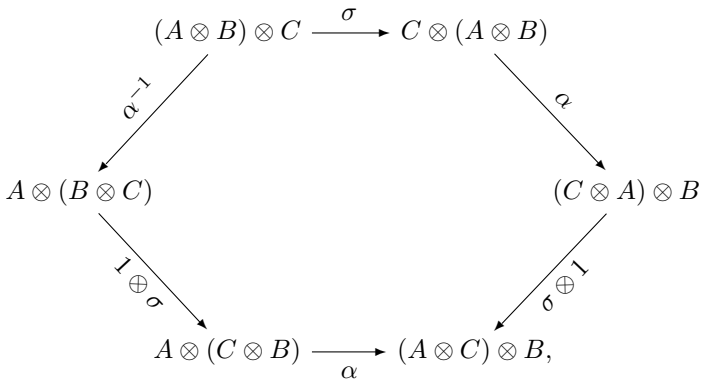
$$\begin{array}{ccc} (A\otimes B)\otimes I & \xrightarrow{\alpha} & A\otimes(B\otimes I) \\ & {\scriptstyle \rho}\searrow \quad \swarrow{\scriptstyle 1\otimes\rho} & \\ & A\otimes B & \end{array} \qquad \begin{array}{ccc} (I\otimes A)\otimes B & \xrightarrow{\alpha} & I\otimes(A\otimes B) \\ & {\scriptstyle \lambda\otimes 1}\searrow \quad \swarrow{\scriptstyle \lambda} & \\ & A\otimes B. & \end{array}$$

Proof. See [JS93]. □

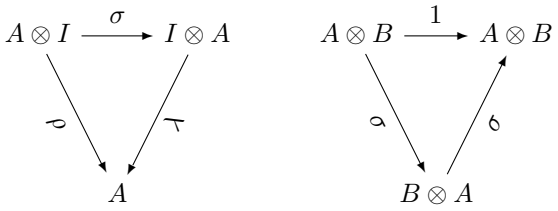
Definition 2.3. A monoidal category is *symmetric* if it has a natural isomorphism

$$\sigma_{A,B} : A \otimes B \to B \otimes A$$

such that



and



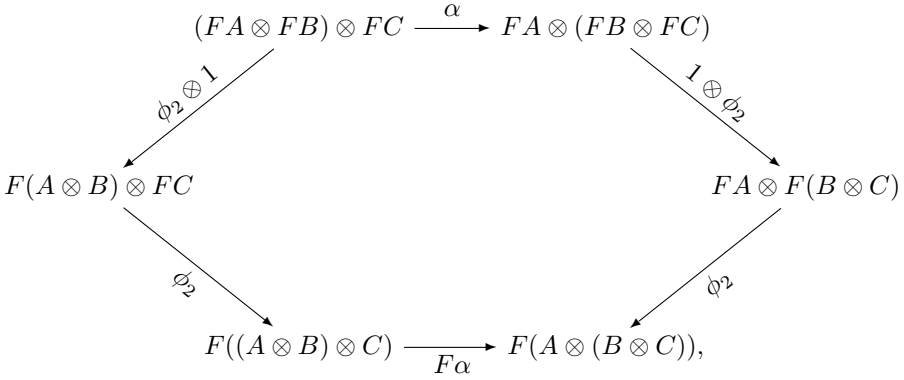
commute.

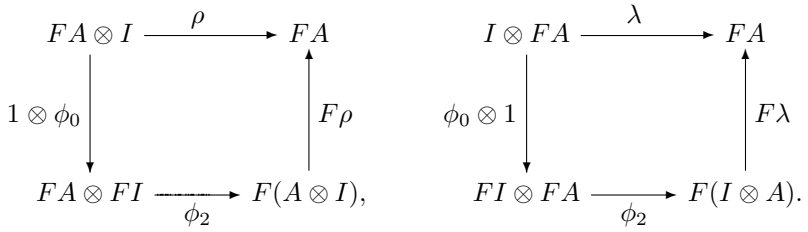
Mac Lane's celebrated coherence theorem states that any formal diagram constructed from the α, ρ, λ and σ will commute.

Definition 2.4. Given monoidal categories $\mathcal{A}$ and $\mathcal{B}$, a *monoidal functor* $F : \mathcal{A} \to \mathcal{B}$ is a functor F and a pair of natural transformations

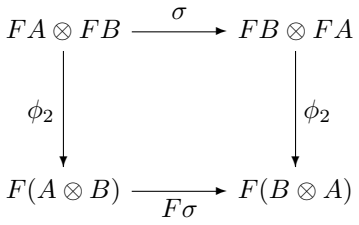
$$\phi_2 : FA \otimes FB \to F(A \otimes B)$$
$$\phi_0 : I \to FI$$

such that the following diagrams commute:

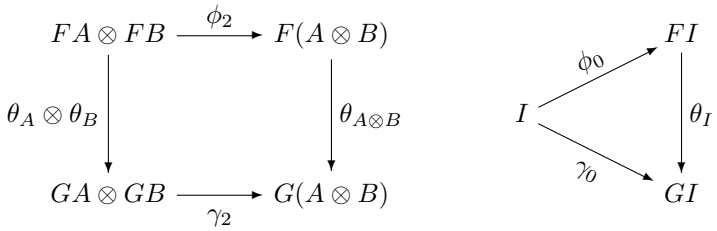




If ϕ_2, ϕ_0 are isomorphisms then F is called *strong*; if they are identities then F is *strict*. A monoidal functor between symmetric monoidal categories is *symmetric* if it commutes with the symmetry as shown by the diagram below.



Definition 2.5. A natural transformation between two monoidal functors $\theta : (F, \phi_2, \phi_0) \Rightarrow (G, \gamma_2, \gamma_0)$ is called *monoidal* if the following diagrams commute for all objects A, B of the underlying category.



The identity natural transformation is evidently monoidal, and the horizontal or vertical composition of two monoidal natural transformations is also monoidal. Hence this definition makes the class of small monoidal categories into a 2-category which we write **Mon**. Since **Mon** is a 2-category, each of its hom-sets $\mathbf{Mon}(\mathcal{A}, \mathcal{B})$ forms a category whose objects are monoidal functors between any two monoidal categories $\mathcal{A}, \mathcal{B}$, and whose arrows are monoidal natural transformations. Note also that $\mathbf{Mon}(\mathcal{A}, \mathcal{A})$ (indeed, any category of endofunctors) is itself strict monoidal, with the tensor given by composition of functors. We write **SMon** for the sub 2-category of symmetric monoidal categories and functors.

Recall that an equivalence of categories $\mathcal{A}$, $\mathcal{B}$ is a pair of functors $F : \mathcal{A} \to \mathcal{B}, G : \mathcal{B} \to \mathcal{A}$ such that both FG and GF are naturally isomorphic to the identity.

Theorem 2.6 (Mac Lane)**.** *Every monoidal category $\mathcal{C}$ is equivalent to some strict monoidal category $\mathcal{A}$ via a strong monoidal functor $G : \mathcal{C} \to \mathcal{A}$ and a strict monoidal functor $F : \mathcal{A} \to \mathcal{C}$.*

The above theorem is sharpened by Joyal and Street [JS93] to state that each monoidal category $\mathcal{A}$ is equivalent to a particular strict monoidal category

$\mathbf{st}(\mathcal{A})$, and further, each monoidal functor $F : \mathcal{A} \to \mathcal{B}$ induces a strict monoidal functor $\mathbf{st}(T) : \mathbf{st}(\mathcal{A}) \to \mathbf{st}(\mathcal{B})$. Hence any diagram of monoidal categories and functors can be replaced with an equivalent diagram of strict monoidal categories and functors. Given this, we will assume without further remark that any monoidal category under discussion is strict whenever convenient to do so, and likewise for functors.

2.2 Duality

Let $\mathcal{C}$ be a monoidal category. Suppose we have maps

$$\eta : I \to B \otimes A$$
$$\epsilon : A \otimes B \to I$$

then say that B is *dual* (or *adjoint*) to A if the composites

$$A \xrightarrow{1_A \otimes \eta} A \otimes B \otimes A \xrightarrow{\epsilon \otimes 1_A} A \tag{2.1}$$

and

$$B \xrightarrow{\eta \otimes 1_B} B \otimes A \otimes B \xrightarrow{1_B \otimes \epsilon} B \tag{2.2}$$

are equal to 1_A and 1_B respectively. In this case we write $(\eta, \epsilon) : A \dashv B$, and call η and ϵ the *unit* and *counit* maps. If $A \dashv B$ and $A' \dashv B'$ then there is a bijection between $\mathcal{C}(A, A')$ and $\mathcal{C}(B', B)$: given $f : A \to A'$, define $g : B' \to B$ by

$$\begin{array}{ccc} B' & \xrightarrow{\eta \otimes 1_{B'}} & B \otimes A \otimes B' \\ {\scriptstyle g}\downarrow & & \downarrow{\scriptstyle 1_B \otimes f \otimes 1_{B'}} \\ B & \xleftarrow[1_B \otimes \epsilon']{} & B \otimes A' \otimes B'. \end{array} \tag{2.3}$$

If $f : A \to A'$ and $g : B' \to B$ are related by this bijection, we say that g is the *dual* of f and write $f \dashv g$.

Proposition 2.7. *If $A \dashv B$ and also $A \dashv C$ then $B \cong C$ and this isomorphism is natural in the following sense: suppose that $A_1 \dashv B_1$, $A_1 \dashv C_1$ and also $A_2 \dashv B_2$, $A_2 \dashv C_2$; further let $f : A_1 \to A_2$. Morphisms $f_B : B_2 \to B_1$ and $f_C : C_2 \to C_1$, dual to f, necessarily exist, and these make the diagram*

$$\begin{array}{ccc} B_2 & \xrightarrow{\cong} & C_2 \\ {\scriptstyle f_B}\downarrow & & \downarrow{\scriptstyle f_C} \\ B_1 & \xrightarrow[\cong]{} & C_1, \end{array} \tag{2.4}$$

commute.

Proof. The isomorphism between B and C is given by the arrow

$$d : B \xrightarrow{\eta_C \otimes 1_B} C \otimes A \otimes B \xrightarrow{1_C \otimes \epsilon_B} C.$$

Let $d' = (1_B \otimes \epsilon_C) \circ (\eta_B \otimes 1_C)$, then

$$\begin{aligned} d' \circ d &= (1_B \otimes \epsilon_C) \circ (\eta_B \otimes 1_C) \circ (1_C \otimes \epsilon_B) \circ (\eta_C \otimes 1_B) \\ &= (1_B \otimes \epsilon_B) \circ (1_B \otimes ((\epsilon_C \otimes 1_A) \circ (1_A \otimes \eta_C)) \otimes 1_B) \circ (\eta_B \otimes 1_B) \\ &= (1_B \otimes \epsilon_B) \circ (\eta_B \otimes 1_B) \\ &= 1_B, \end{aligned}$$

where the second line follows by the functoriality of the tensor. Similarly $d \circ d' = 1_C$. To establish the naturality condition (2.4) note that

$$\begin{aligned} f_C \circ d_2 &= (1_{C_1} \otimes \epsilon_{C_2}) \circ (1_{C_1} \otimes f \otimes 1_{C_2}) \circ (\eta_{C_1} \otimes 1_{C_2}) \\ &\qquad \circ (1_{C_2} \otimes \epsilon_{B_2}) \circ (\eta_{C_2} \otimes 1_{B_2}) \\ &= (1_{C_1} \otimes \epsilon_{B_2}) \circ (1_{C_1} \otimes ((\epsilon_{C_2} \otimes 1_{C_2}) \circ (1_{A_2} \otimes \eta_{C_2})) \otimes 1_{B_2}) \\ &\qquad \circ (1_{C_1} \otimes f \otimes 1_{B_2}) \circ (\eta_{C_1} \otimes 1_{B_2}) \\ &= (1_{C_1} \otimes \epsilon_{B_2}) \circ (1_{C_1} \otimes f \otimes 1_{B_2}) \circ (\eta_{C_1} \otimes 1_{B_2}). \end{aligned}$$

A similar calculation gives

$$d_1 \circ f_B = (1_{C_1} \otimes \epsilon_{B_2}) \circ (1_{C_1} \otimes f \otimes 1_{B_2}) \circ (\eta_{C_1} \otimes 1_{B_2})$$

as required. □

Proposition 2.8. *Strong monoidal functors preserve duals. More explicitly, if $F : \mathcal{A} \to \mathcal{B}$ is a strong monoidal functor, then $A \dashv B$ implies $FA \dashv FB$ and $f \dashv g$ implies $Ff \dashv Fg$.*

Proof. Suppose $(\eta, \epsilon) : A \dashv B$ in $\mathcal{A}$, and let $F = (F, \phi_2, \phi_0)$ be a strong monoidal functor $\mathcal{A} \to \mathcal{B}$. Define

$$\begin{aligned} \eta^F &= I \xrightarrow{\phi_0} FI \xrightarrow{F\eta} F(B \otimes A) \xrightarrow{\phi_2^{-1}} FB \otimes FA \\ \epsilon^F &= FA \otimes FB \xrightarrow{\phi_2} F(A \otimes B) \xrightarrow{F\epsilon} FI \xrightarrow{\phi_0^{-1}} I \end{aligned}$$

To show that $(\eta^F, \epsilon^F) : FA \dashv FB$ we need to show that the equations

$$\begin{aligned} 1_{FA} &= (\epsilon^F \otimes 1_A) \circ (1_A \otimes \eta^F), \\ 1_{FB} &= (1_B \otimes \epsilon^F) \circ (\eta^F \otimes 1_B) \end{aligned}$$

hold in $\mathcal{B}$. Figure 2.1 shows a diagrammatic proof of the first equation. The lower edge of the diagram is the expanded definition of the right hand side of the equation: the end triangles and the middle diamond commute due to the coherence conditions of monoidal functors; the upper triangle commutes because $A \dashv B$ in $\mathcal{A}$; and the remaining quadrilaterals commute due to the naturality of ϕ_2. The proof of the second equation is essentially the same.

To show that $f \dashv g$ implies $Ff \dashv Fg$ for some arrows $f : A_0 \to A_1, g : B_1 \to B_0$ we must prove that

$$Ff = (\epsilon_0^F \otimes 1_{FA_1}) \circ (1_{FA_0} \otimes Fg \otimes 1_{FA_1}) \circ (1_{FA_0} \otimes \eta_1^F).$$

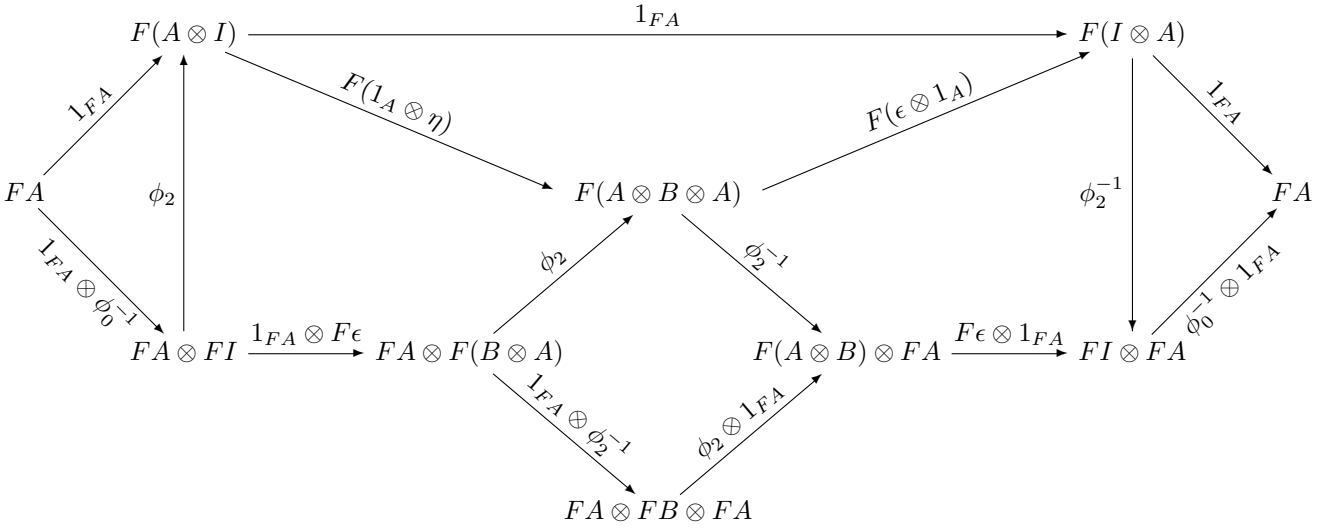


Figure 2.1: Diagrammatic proof of proposition 2.8

Since $f \dashv g$ we have also that

$$Ff = F(\epsilon_0 \otimes 1_{A_1}) \circ F(1_{A_0} \otimes g \otimes 1_{A_1}) \circ F(1_{A_0} \otimes \eta_1).$$

Establishing the equality of these two expressions involves little more than the coherence conditions on F, and can tackled with a very similar proof to that shown in figure 2.1. □

2.3 Compact Closed Categories

Definition 2.9. A symmetric monoidal category is called *compact closed* if every object A has a chosen dual $(\eta_A, \epsilon_A) : A \dashv A^*$.

As we have already shown, duals are unique up to isomorphism; the choice of a particular dual makes the compact closed structure equational. Although strong monoidal functors preserve duals, they do so only up to isomorphism. In order to preserve this equational structure we make the following definition.

Definition 2.10. A functor $F : \mathcal{A} \to \mathcal{B}$ between two compact closed categories is *compact closed* if it is strictly monoidal and preserves the assigned duals exactly:

$$F(A^*) = (FA)^* \qquad F\eta_A = \eta_{FA} \qquad F\epsilon_A = \epsilon_{FA}.$$

The class of all small compact closed categories and compact closed functors between them forms a category **ComCl**. The forgetful functor $U : \mathbf{ComCl} \to \mathbf{Cat}$ has a left adjoint F, giving the free compact closed category generated by a category. We postpone discussion of the free construction until Section 2.8. For the moment we concentrate on the elementary properties of compact closed categories, and in particular the role of the duality.

Proposition 2.11. $(-)^*$ *defines a contravariant functor with* $f \dashv f^*$.

Proof. We have $1_A^* = 1_{A^*}$ immediately from the definition of dual, and $(f \circ g)^* = g^* \circ f^*$ follows from a routine calculation, very similar to the proof of proposition 2.7. □

Lemma 2.12. *In a compact closed category* $\mathcal{C}$ *we have the following dual pairs:*

1. $A \otimes B \dashv B^* \otimes A^*$;
2. $I \dashv I$;
3. $A^* \dashv A$.

Proof. We give the definitions of the unit and counit maps in each case.

1. $(\eta_1, \epsilon_2) : A \otimes B \dashv B^* \otimes A^*$ via

$$\begin{aligned} \eta_1 &= I \xrightarrow{\eta_B} B^* \otimes B \xrightarrow{1_{B^*} \otimes \eta_A \otimes 1_B} B^* \otimes A^* \otimes A \otimes B, \\ \epsilon_1 &= A \otimes B \otimes B^* \otimes A^* \xrightarrow{1_A \otimes \epsilon_B \otimes 1_{A^*}} A \otimes A^* \xrightarrow{\epsilon_A} I. \end{aligned}$$

2. $(\eta_2, \epsilon_2) : I \dashv I$ via

$$\begin{aligned} \eta_2 &= I \xrightarrow{\lambda_I^{-1}} I \otimes I, \\ \epsilon_3 &= I \otimes I \xrightarrow{\lambda_I} I. \end{aligned}$$

3. $(\eta_3, \epsilon_3) : A^* \dashv A$ via

$$\begin{aligned} \eta_3 &= I \xrightarrow{\eta_A} A^* \otimes A \xrightarrow{\sigma} A \otimes A^*, \\ \epsilon_3 &= A^* \otimes A \xrightarrow{\sigma} A \otimes A^* \xrightarrow{\epsilon_A} I. \end{aligned}$$

Verifying that equations (2.1) and (2.2) hold is routine. □

By proposition 2.7, duals are naturally isomorphic, hence we have natural isomorphisms

$$\begin{aligned} u : (A \otimes B)^* &\cong B^* \otimes A^* \\ v : I^* &\cong I \\ w : A^{**} &\cong A, \end{aligned}$$

which lead to the following corollary.

Proposition 2.13. *The functor* $(\cdot)^* : \mathcal{C}^{op} \to \mathcal{C}$ *is strong monoidal, and an equivalence of categories.*

Since we have the equivalence between $\mathcal{C}$ and $\mathcal{C}^{\mathrm{op}}$, any statement about some arrow applies equally well to its dual. In particular, results concerning units translate directly into results about counits and vice-versa.

A compact closed category which, in addition to being strictly monoidal, has all of the isomorphisms u, v, w equal to the identity is called a *strict compact*

closed category. Kelly and Laplaza [KL80] show that any compact closed category is equivalent to a strict one, hence we will take the isomorphisms above to be equalities whenever convenient.

The unit maps constructed in Lemma 2.12 allow an equational characterisation of $\eta_{A\otimes B}, \eta_I$ and η_{A^*} in terms of the isomorphisms u, v, and w. In a strict compact closed category we have equalities $\eta_1 = \eta_{A\otimes B}$, $\eta_2 = \eta_I$ and $\eta_3 = \eta_{A^*}$.

Lemma 2.14 (Kelly-Laplaza)**.** *The following are equivalent:*

$$\begin{aligned}
(1_{A^*} \otimes f) \circ \eta_A &= (f^* \otimes 1_B) \circ \eta_B; \\
\epsilon_B \circ (f \otimes 1_{B^*}) &= \epsilon_A \circ (1_A \otimes f^*); \\
f &= (1_B \otimes \epsilon_A) \circ (1_B \otimes f^* \otimes 1_A) \circ (\eta_B \otimes 1_A); \\
f^* &= (1_{A^*} \otimes \epsilon_B) \circ (1_{A^*} \otimes f \otimes 1_{B^*}) \circ (\eta_A \otimes 1_{B^*}).
\end{aligned}$$

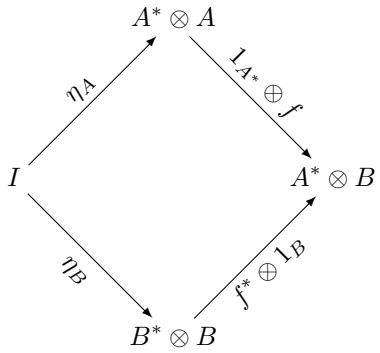
Proof. See [KL80]. □

Note that the last of these is the defining equation of f^* (equation (2.3)), which leads directly to the following.

Proposition 2.15. *In a compact closed category the units and counits define dinatural transformations (see [GSS91])*

$$\begin{aligned}
\eta &: I \Rightarrow ((-)^* \otimes -) \\
\epsilon &: (- \otimes (-)^*) \Rightarrow I.
\end{aligned}$$

Proof. Treating the case of the unit only, it must be shown that



which is exactly the first equation of lemma 2.14. The second equation gives the dinaturality of the counit. □

Proposition 2.16 (Joyal-Street)**.** *If $\mathcal{C}$ is compact closed then $\boldsymbol{Mon}(\mathcal{C}, \mathcal{A})$ is a groupoid. More explicitly, $\alpha_A : FA \to GA$ is invertible with $\alpha_A^{-1} = (\alpha_{A^*})^*$, up to a canonical isomorphism.*

Proof. We take both categories and functors as strict, and assume for simplicity of exposition that $(FA^*)^* = FA$ and $(GA^*)^* = GA$, since in all cases they are canonically isomorphic.

By definition, $(\alpha_{A^*})^*$ is

$$GA \xrightarrow{1_{GA}\otimes F\eta} GA \otimes FA^* \otimes FA \xrightarrow{1_{GA}\otimes\alpha_{A^*}\otimes 1_{FA}} GA \otimes GA^* \otimes FA \xrightarrow{G\eta\otimes 1_{FA}} FA.$$

Since α is a monoidal natural transformation, we have

$$\alpha_{A^* \otimes A} = (\alpha_{A^*} \otimes 1_{GA}) \circ (1_{FB} \otimes \alpha_A)$$

and

$$\alpha_{A^* \otimes A} \circ F\eta = G\eta.$$

Hence

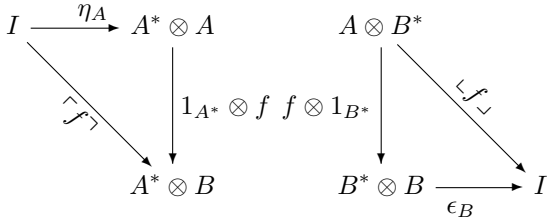
$$\begin{aligned} \alpha_A \circ (\alpha_{A^*})^* &= (G\epsilon_A \otimes \alpha_A) \circ (1_{GA} \otimes \alpha_{A^*} \otimes 1_{FA}) \circ (1_{GA} \otimes F\eta_A) \\ &= (G\epsilon_A \otimes 1_{GA}) \circ (1_{GA} \otimes \alpha_{A^* \otimes A}) \circ (1_{GA} \otimes F\eta_A) \\ &= (G\epsilon_A \otimes 1_{GA}) \circ (1_{GA} \otimes G\eta_A) \\ &= 1_{GA} \end{aligned}$$

and similarly $(\alpha_{A^*})^* \circ \alpha_A = 1_{FA^*}$; so $\alpha_A^{-1} = (\alpha_{A^*})^*$ as required. □

2.4 Names and Conames

The duality of a compact closed category gives a particularly strong form of monoidal closure. Every arrow in the category has a *point* which represents it, and dually a *copoint*. These representatives, the names and conames, will be crucial to our treatment of entangled quantum states.

Definition 2.17. Let $f : A \to B$ in a compact closed category $\mathcal{C}$. Define the *name* and *coname* of f to be the maps $\ulcorner f \urcorner : I \to A^* \otimes B$ and $\llcorner f \lrcorner : A \otimes B^* \to I$ which are defined by the diagrams below.



An immediate consequence of this definition is the isomorphism of hom-sets

$$\mathcal{C}(I, A^* \otimes B) \cong \mathcal{C}(A, B) \cong \mathcal{C}(A \otimes B^*, I).$$

Lemma 2.18 (absorption)**.** *Let*

$$D \xrightarrow{h} A \xrightarrow{f} B \xrightarrow{g} C;$$

then we have

$$(1_{A^*} \otimes g) \circ \ulcorner f \urcorner = \ulcorner g \circ f \urcorner,$$

and

$$(h^* \otimes 1_B) \circ \ulcorner f \urcorner = \ulcorner f \circ h \urcorner.$$

Proof. The first equation is immediate from the definition. In the second we have

$$\begin{aligned}(h^* \otimes 1_B) \circ \ulcorner f \urcorner &= (h^* \otimes 1_B) \circ (1_{A^*} \otimes f) \circ \eta_A \\ &= (1_{D^*} \otimes f) \circ (h^* \otimes 1_A) \circ \eta_A \\ &= (1_{A^*} \otimes f) \circ (1_{A^*} \otimes h) \circ \eta_D\end{aligned}$$

by definition, functoriality of $\otimes$, and dinaturality of η, respectively. □

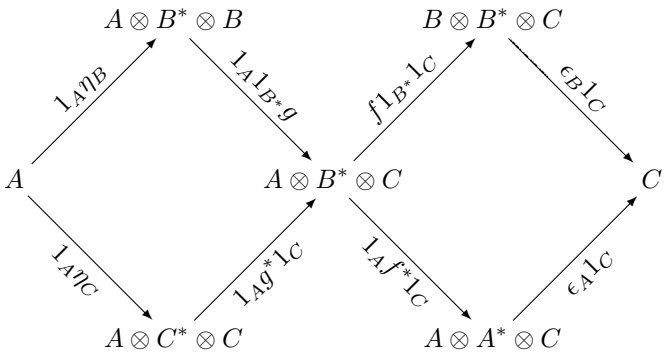
Lemma 2.19 (Compositionality)**.** *Given*

$$A \xrightarrow{f} B \xrightarrow{g} C;$$

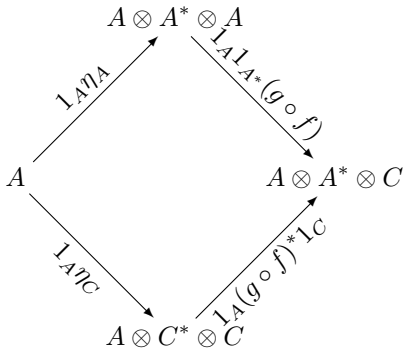
we have

$$(\llcorner f \lrcorner \otimes 1_C) \circ (1_A \otimes \ulcorner g \urcorner) = g \circ f.$$

Proof. Since η and ϵ are dinatural we have the following commuting squares (tensor written as juxtaposition on the arrow labels):



We have $f^* \circ g^* = (g \circ f)^*$ and hence, again by dinaturality,



commutes. Taking these together we have

$$\begin{aligned}&(\llcorner f \lrcorner \otimes 1_C) \circ (1_A \otimes \ulcorner g \urcorner) \\ &\quad = (\epsilon_A \otimes 1_C) \circ (1_A \otimes 1_{A^*} \otimes (g \circ f)) \circ (1_A \otimes \eta_A) \\ &\quad = (g \circ f) \circ (\epsilon_A \otimes 1_A) \circ (1_A \otimes \eta_A) \\ &\quad = g \circ f\end{aligned}$$

as required. □

Lemma 2.20 (Compositional Cut)**.** *Suppose we have*

$$A \xrightarrow{f} B \xrightarrow{g} C \xrightarrow{h} D;$$

then

$$(1_{A^*} \otimes \llcorner g \lrcorner \otimes 1_D) \circ (\ulcorner f \urcorner \otimes \ulcorner h \urcorner) = \ulcorner h \circ g \circ f \urcorner.$$

Proof. By the functoriality of the tensor

$$(1_{A^*} \otimes \llcorner g \lrcorner \otimes 1_D) \circ (\ulcorner f \urcorner \otimes \ulcorner h \urcorner) = ([(1_{A^*} \otimes \llcorner g \lrcorner) \circ (\ulcorner f \urcorner \otimes 1_{C^*})] \otimes 1_D) \circ \ulcorner h \urcorner.$$

By the composition lemma $(1_{A^*} \otimes \llcorner g \lrcorner) \circ (\ulcorner f \urcorner \otimes 1_{C^*}) = (g \circ f)^*$ hence

$$(1_{A^*} \otimes \llcorner g \lrcorner \otimes 1_D) \circ (\ulcorner f \urcorner \otimes \ulcorner h \urcorner) = ((g \circ f)^* \otimes 1_D) \circ \ulcorner h \urcorner$$

from which the result follows by absorption. □

We can also define partial versions of the name and coname; essentially currying and uncurrying.

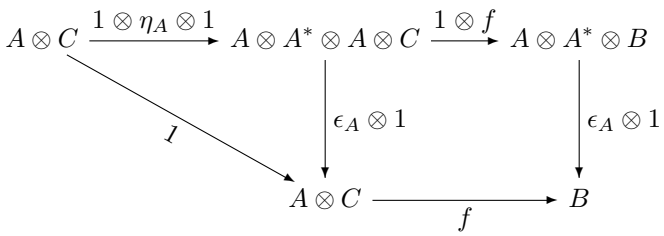
Lemma 2.21 (Partial Names and Conames)**.** *In any compact closed category we have the following isomorphisms:*

$$\mathcal{C}(A \otimes C, B) \cong \mathcal{C}(A, C^* \otimes B) \tag{2.5}$$
$$\mathcal{C}(A, C \otimes B) \cong \mathcal{C}(A \otimes C^*, B) \tag{2.6}$$

Proof. Since the two isomorphisms are dual, we prove only the first. Define $F : \mathcal{C}(A \otimes C, B) \to \mathcal{C}(A, C^* \otimes B)$ and $G : \mathcal{C}(A, C^* \otimes B) \to \mathcal{C}(A \otimes C, B)$ by

$$F : f \mapsto (1 \otimes f) \circ (\eta_A \otimes 1)$$
$$G : g \mapsto (\epsilon_A \otimes 1) \circ (1 \otimes g)$$

Their composition gives $GFf = (\epsilon_A \otimes 1) \circ (1 \otimes f) \circ (1 \otimes \eta_A 1)$ from which



and hence $GF = \mathrm{Id}$. Similarly $\mathrm{Id} = FG$, which establishes the isomorphism. □

Equation (2.5) essentially states that compact closed categories are indeed closed with $B^A = A^* \otimes B$. Since $A^* \otimes I \cong A^*$ this gives immediately the following.

Corollary 2.22. *Compact closed categories are $*$-autonomous[Bar79].*

2.5 Strong Compact Closure

The usual von Neumann account of quantum mechanics takes place in the category of Hilbert spaces: for the purposes of quantum computation we restrict to the finite dimensional case and operate in the category **FDHilb**. This category is, indeed, compact closed. However the language of compact closed categories does not offer a complete description of the Hilbert space structure. The missing component is the inner product $\langle \cdot \mid \cdot \rangle : H \times H \to \mathbb{C}$. Without it, crucial parts of quantum mechanics cannot be represented. To address this shortcoming, Abramsky and Coecke [AC04] introduced *strong compact closed categories*.

In a Hilbert space, the inner product is regarded as primitive and used to define the adjoint $f^\dagger$ of a linear map f. The adjoint so-defined is a contravariant involutive functor on **FDHilb**. Since the vectors of the space H can themselves be thought of as linear maps $I \to H$, the adjoint can be used to define the inner product. Recognising how the adjoint interacts with the usual dual leads to the definition of strongly compact closed category. We use here the refined definition of [AC05].

Definition 2.23. A *strongly compact closed category* is a symmetric monoidal category $\mathcal{C}$ equipped with the following data:

1. an involutive monoidal assignment $A \mapsto A^*$ on objects;

2. an involutive, contravariant, strict monoidal functor $(\cdot)^\dagger$ such that $A^\dagger = A$ on all objects A, and for each of the monoidal structure isomorphisms $\sigma, \alpha, \rho, \lambda$ we have $f^\dagger = f^{-1}$;

3. to each object A a map $\eta_A : I \to A^* \otimes A$ such that the equations

$$\eta_{A^*} = \sigma_{A^*,A} \circ \eta_A \tag{2.7}$$

$$1_A = A \xrightarrow{1_A \otimes \eta_A} A \otimes A^* \otimes A \xrightarrow{(\eta_A^\dagger \circ \sigma_{A,A^*}) \otimes 1_A} A \tag{2.8}$$

   hold for all A.

Note that equation (2.8) is simply the first defining equation (2.1) of an ordinary compact closed category with $\epsilon_A = (\eta_A^\dagger \circ \sigma_{A,A^*})$. Performing the same substitution upon the dual condition (2.2) yields

$$f = A^* \xrightarrow{\eta_A \otimes 1_{A^*}} A^* \otimes A \otimes A^* \xrightarrow{1_{A^*} \otimes (\eta_A^\dagger \circ \sigma_{A,A^*})} A^*$$

from whence

$$f^\dagger = A^* \xrightarrow{1_{A^*} \otimes (\sigma_{A^*,A} \circ \eta_A)} A^* \otimes A \otimes A^* \xrightarrow{\eta_A^\dagger \otimes 1_{A^*}} A^*$$

and since $A = A^{**}$, equations (2.7) and (2.8) yield $f^\dagger = 1_A = 1_A^\dagger = f^{\dagger\dagger} = f$, so strongly compact closed categories are indeed compact closed. Note that while $A^{**} = A$, a strong compact closed category is not necessarily strictly compact closed; though in general we shall assume this.

2.6 Trace

The notion of a *trace* on a monoidal category was introduced by Joyal, Street and Verity in [JSV96] and has since found wide application. Here we review some facts about the trace in the context of compact closed categories. This definition is that of [AHS02], which is equivalent to that of the original [JSV96].

Definition 2.24. A *trace* on a symmetric monoidal category $\mathcal{C}$ is a family of functions $\mathrm{Tr}^C_{A,B} : \mathcal{C}(A \otimes C, B \otimes C) \to \mathcal{C}(A, B)$ which obeys the following axioms.

Naturality in A Given $f : A \otimes C \to B \otimes C$ and $g : A' \to A$ then

$$\mathrm{Tr}^C_{A,B}(f) \circ g = \mathrm{Tr}^C_{A,B}(f \circ (g \otimes 1_C)).$$

Naturality in B Given $f : A \otimes C \to B \otimes C$ and $h : B \to B'$ then

$$g \circ \mathrm{Tr}^C_{A,B}(f) = \mathrm{Tr}^C_{A,B}((g \otimes 1_C) \circ f).$$

Dinaturality in C Given $f : A \otimes C \to B \otimes C'$ and $k : C' \to C$ then

$$\mathrm{Tr}^C_{A,B}((1_B \otimes k) \circ f) = \mathrm{Tr}^{C'}_{A,B}(f \circ (1_A \otimes k)).$$

Vanishing Given $f : A \otimes I \to B \otimes I$ and $g : A \otimes C \otimes D \to B \otimes C \otimes D$ then

$$\begin{aligned} \mathrm{Tr}^I_{A,B}(f) &= f; \\ \mathrm{Tr}^{C \otimes D}_{A,B}(g) &= \mathrm{Tr}^C_{A,B}(\mathrm{Tr}^D_{A \otimes C, B \otimes C}(g)). \end{aligned}$$

Superposing Given $f : A \otimes C \to B \otimes C'$ and $g : D \to D'$ then

$$g \otimes \mathrm{Tr}^C_{A,B}((1_B \otimes k) \circ f) = \mathrm{Tr}^{C'}_{D \otimes A, D' \otimes B}(f).$$

Yanking $\mathrm{Tr}^A_{A,A}(\sigma_{A,A}) = 1_A$.

A symmetric monoidal category is called *traced* if a trace exists for all objects A, B, C.

Proposition 2.25 (Joyal-Street-Verity)**.** *Every compact closed category is traced, via the* canonical trace

$$\mathrm{Tr}^C_{A,B}(f) = A \xrightarrow{1_A \otimes \eta_{C^*}} A \otimes C \otimes C^* \xrightarrow{f \otimes 1_{C^*}} B \otimes C \otimes C^* \xrightarrow{1_B \otimes \epsilon_C} B.$$

Conversely every traced monoidal category has canonical extension to a compact closed category via the Int construction.

Definition 2.26. Let $\mathcal{C}$ be a traced symmetric monoidal category; we define a category Int$\mathcal{C}$ as follows. The objects of Int$\mathcal{C}$ are pairs (A, B) of objects A, B of $\mathcal{C}$. An arrow $f : (A, B) \to (C, D)$ of Int$\mathcal{C}$ is an arrow $f : A \otimes D \to C \otimes B$. The composition of $f : (A, B) \to (C, D)$ and $g : (C, D) \to (E, F)$ is Int$\mathcal{C}$ is given by the trace $\mathrm{Tr}^D(h) : A \otimes F \to E \otimes B$ in $\mathcal{C}$ where h is the composite

$$A \otimes F \otimes D \xrightarrow{\cong} A \otimes D \otimes F \xrightarrow{f \otimes 1} C \otimes B \otimes F \xrightarrow{\cong} C \otimes F \otimes B \xrightarrow{g \otimes 1} E \otimes D \otimes B \xrightarrow{\cong} E \otimes B \otimes D.$$

The identity $1_{(A,B)}$ in Int$\mathcal{C}$ is the arrow $1_A \otimes 1_B$ in $\mathcal{C}$.

Remark. The original Int construction is developed in a non-symmetric setting; the definitions presented here are modified accordingly.

We omit the description of the tensor and compact closed structures; see [JSV96].

Proposition 2.27. Int$\mathcal{C}$ *is a compact closed category and the assignments* $N(A) = (A, I)$ *and* $N(f) = f$ *define a traced, strong monoidal, fully faithful functor* $N : \mathcal{C} \to \mathrm{Int}\mathcal{C}$.

Theorem 2.28 (Joyal-Street-Verity)**.** *Let* $\mathcal{C}$ *be a traced symmetric monoidal category and* $\mathcal{D}$ *a compact closed category. For every trace preserving functor* $F : \mathcal{C} \to \mathcal{D}$ *there exists a compact closed functor* $K : \mathrm{Int}\mathcal{C} \to \mathcal{D}$ *which is unique up to monoidal natural isomorphism with* $KN \cong F$. *Further, the inclusion of* **ComCl** *into* **TraMon** *has a left biadjoint with unit having component at* $\mathcal{C}$ *given by* $N : \mathcal{C} \to \mathrm{Int}\mathcal{C}$.

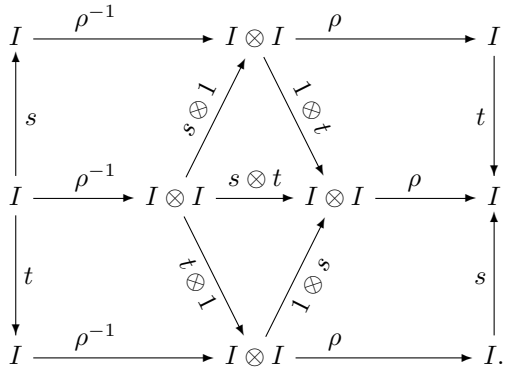
Remark. The essential content of this result is already contained in our Lemma 2.21.

2.7 Scalars and Loops

Definition 2.29. In any monoidal category $\mathcal{C}$ the endomorphisms of the neutral element $\mathcal{C}(I, I)$ are called the *scalars*.

Lemma 2.30. *The scalars form a commutative monoid with respect to composition.*

Proof. Let $s, t \in \mathcal{C}(I, I)$; then



□

Corollary 2.31. *For scalars* s, t *the composite*

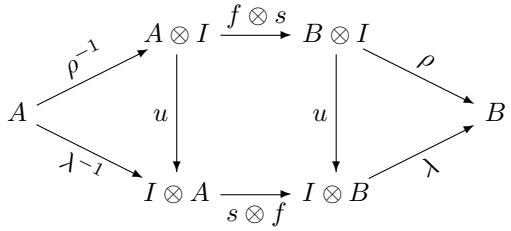
$$I \cong I \otimes I \xrightarrow{s \otimes t} I \otimes I \cong I$$

is equal to $s \circ t = t \circ s$.

Definition 2.32. Let $\mathcal{C}$ be a monoidal category. Given a scalar s and some arrow $f : A \to B$ define a scalar multiplication $s \bullet f$ by the composition:

$$A \xrightarrow{\rho^{-1}} A \otimes I \xrightarrow{f \otimes s} B \otimes I \xrightarrow{\rho} B.$$

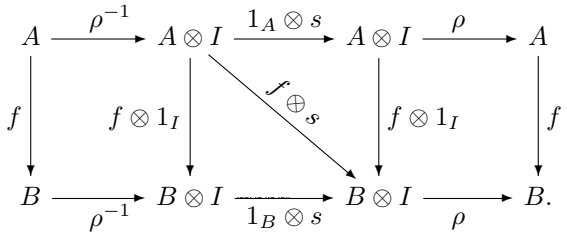
Remark. We could have defined $s \bullet f$ equivalently by multiplication on the left rather than on the right as above. Note that $u := \lambda^{-1} \circ \rho$ is a natural isomorphism $(- \otimes I) \Rightarrow (I \otimes -)$, so the following diagram commutes



and hence the two definitions coincide.

Lemma 2.33. *Each scalar s determines a natural transformation $Id \Rightarrow Id$ such that $s \bullet f = f \circ s_A = s_B \circ f$.*

Proof. The top and bottom edges define s_A and s_B respectively:



The outer squares commute due to naturality of ρ, and the middle due to the functoriality of the tensor. Hence s defines a natural transformation. Note that the middle path from A to B is the definition of $s \bullet f$. □

Corollary 2.34. *The following are immediate.*

1. $s \bullet (t \bullet f) = (s \circ t) \bullet f$
2. $(s \bullet f) \circ (t \bullet g) = (s \circ t) \bullet (f \circ g)$
3. $(s \bullet f) \otimes (t \bullet g) = (s \circ t) \bullet (f \otimes g)$

Definition 2.35. In a compact closed category $\mathcal{C}$ define the *dimension* of an object A, to be the following composite:

$$\dim_A = I \xrightarrow{\eta_A} A^* \otimes A \xrightarrow{\sigma} A \otimes A^* \xrightarrow{\epsilon_A} I.$$

Lemma 2.36. *If $f : A \xrightarrow{\cong} B$ is an isomorphism in a compact closed category $\mathcal{C}$, then $dim_A = dim_B$.*

Proof. Since $(\cdot)^*$ is a functor we have $(f \circ g)^* = g^* \circ f^*$. Then

$$\begin{aligned}
\dim_A &= I \xrightarrow{\eta_A} A^* \otimes A \xrightarrow{\epsilon_{A^*}} I \\
&= I \xrightarrow{\eta_A} A^* \otimes A \xrightarrow{1_{A^*} \otimes f} A^* \otimes B \xrightarrow{1_{A^*} \otimes f^{-1}} A^* \otimes A \xrightarrow{\epsilon_{A^*}} I \\
&= I \xrightarrow{\eta_B} B^* \otimes B \xrightarrow{f^* \otimes 1_B} A^* \otimes B \xrightarrow{(f^{-1})^* \otimes 1_B} B^* \otimes B \xrightarrow{\epsilon_{B^*}} I \\
&= I \xrightarrow{\eta_B} B^* \otimes B \xrightarrow{\epsilon_{B^*}} I \\
&= \dim_B.
\end{aligned}$$

□

In a free compact closed category on a discrete category (i.e. only identity morphisms) then the only scalars are dimensions of the objects.

Lemma 2.37. *In a strongly compact closed category let $f : A \to B$ be unitary; then $dim_A = (\ulcorner f \urcorner)^\dagger \circ \ulcorner f \urcorner$.*

Proof.

$$\begin{aligned}(\ulcorner f \urcorner)^\dagger \circ \ulcorner f \urcorner &= (\eta_A)^\dagger \circ (1_{A^*} \otimes f^\dagger) \circ (1_{A^*} \otimes f) \circ \eta_A \\ &= \epsilon_{A^*} \circ \eta_A.\end{aligned}$$

□

Note that for arbitrary maps, $|f| := (\ulcorner f \urcorner)^\dagger \circ \ulcorner f \urcorner$ gives the Hilbert-Schmidt norm in **FDHilb**; see [Coe05]. Indeed for any pair of maps $f, g : A \to B$, we can define the Hilbert-Schmidt inner product

$$\langle f \mid g \rangle := (\ulcorner f \urcorner)^\dagger \circ \ulcorner g \urcorner$$

which for $\psi, \phi : I \to A$ agrees with the inner product given earlier, to wit:

$$\langle \psi \mid \phi \rangle = \psi^\dagger \circ \phi$$

Recall that any compact closed category admits a canonical partial trace [JSV96] $\mathrm{Tr}_B^{A,C}(f) : A \to C$ defined by

$$A \xrightarrow{1_A \otimes \eta_{B^*}} A \otimes B \otimes B^* \xrightarrow{f \otimes 1_{B^*}} C \otimes B \otimes B^* \xrightarrow{1_C \otimes \epsilon_A} C.$$

This can be extended to a full trace over any endomorphism $f : A \to A$ by

$$\mathrm{Tr}(f) = \mathrm{Tr}_A^{I,I}(\rho \circ f \circ \rho^{-1}).$$

Again, for **FDHilb**, this coincides with the usual trace, given by summing the diagonal elements of a matrix representation of f.

2.8 Free Construction

Cyclic structures play an important role in the theory of compact closed categories and, in particular, they give rise to the scalars in free constructions.

Define the set of *endomorphisms* $E(\mathcal{A})$ by the disjoint union

$$E(\mathcal{A}) = \sum_{A \in |\mathcal{A}|} \mathcal{A}(A, A),$$

and let the set of *loops* $[\mathcal{A}]$ be the quotient of $E(\mathcal{A})$ generated by the relation $f \circ g \sim g \circ f$ whenever $A \xrightarrow{f} B \xrightarrow{g} A$. Let $\tau : E(\mathcal{A}) \to [\mathcal{A}]$ be the canonical map onto the loops, and for each endomorphism f write $[f]$ for its image under τ.

The key theorem is the following of [KL80].

Theorem 2.38. *Let*

$$T : \mathcal{A}^{op} \times \mathcal{A} \times \mathcal{A}^{op} \times \mathcal{A} \times \cdots \times \mathcal{A}^{op} \times \mathcal{A} \longrightarrow \mathcal{B}$$

be a functor of $2n$ *variables, let* K *and* L *be objects of* $\mathcal{B}$ *and let* $\alpha : K \Rightarrow T$ *and* $\beta : T \Rightarrow L$ *be natural transformations with typical components*

$$\alpha : K \longrightarrow T(A_1, A_1, A_2, A_2, A_3, \ldots, A_{n-1}, A_n, A_n) \,, \tag{2.9}$$
$$\beta : T(B_1, B_2, B_2, B_3, \ldots, B_{n-1}, B_n, B_n, B_1) \longrightarrow L \,; \tag{2.10}$$

given maps

$$B_1 \xrightarrow{f_1} A_1 \xrightarrow{f_2} B_2 \xrightarrow{f_3} A_2 \xrightarrow{f_4} B_3 \cdots B_n \xrightarrow{f_{2n-1}} A_n \xrightarrow{f_{2n}} B_1$$

the composite of (2.9), $T(f_1, f_2, f_3, \ldots, f_{2n-1}, f_n)$ *and (2.10), depends only on* $[f_{2n} f_{2n-1} \cdots f_2 f_1]$ *so that* α *and* β *give rise to a function* $[\mathcal{A}] \to \mathcal{B}(K, L)$.

Taking $\mathcal{A} = \mathcal{B}$, $K = L = I$, $\alpha = \eta$ and $\beta = \epsilon$ gives a ready source of scalars in any compact closed category $\mathcal{A}$; indeed this is the dimension map given in Definition 2.35. If the category is freely constructed these are the *only* non-trivial scalars. This is a consequence of the more general coherence theorem of Kelly and Laplaza. Before stating the theorem we must introduce some additional terminology, which will be also be required in later chapters.

Definition 2.39. A *signed set* S is a function from a carrier set $|S|$ to the set $\{+, -\}$. Given signed sets R and S, let R^* denote the signed set with the opposite signing to R; let $R \otimes S$ be the disjoint union of R and S, such that $|R \otimes S| = |R| + |S|$.

Definition 2.40. An *involution* is a category which is a coproduct of copies of the category $\mathbf{2}$. Given an involution σ, its object set $|\sigma|$ can form a signed set by assigning $-$ to the source and $+$ to the target of each arrow of $\mathbf{2}$. Call σ an *involution on the signed set* S when this signing agrees with that of S.

Given some category $\mathcal{A}$, we can construct the free compact closed category generated by $\mathcal{A}$, which we call $F\mathcal{A}$. The objects of the $F\mathcal{A}$ are constructed from those of $\mathcal{A}$ by repeated application of the functors $- \otimes -$, $(-)^*$ and the constant I. This characterisation may be used to inductively construct a signed set $S(X)$ corresponding to each object X of $F\mathcal{A}$. Let

$$\begin{aligned} S(I) &= \emptyset \,, \\ S(X \otimes Y) &= S(X) \otimes S(Y) \,, \\ S(X^*) &= S(X)^* \,, \\ S(A) &= \{A \mapsto +\} \qquad \text{if } A \text{ is an object of } \mathcal{A} \,. \end{aligned}$$

The basic structure of arrows in $F\mathcal{A}$ depends upon involutions on the signed sets generated by its objects.

Theorem (Kelly-Laplaza)**.** *Let* $\mathcal{A}$ *be a category; each arrow* $f : A \to B$ *of the free compact closed category generated by* $\mathcal{A}$ *is completely described by the following data:*

1. *An involution* σ *on* $S(A^* \otimes B)$*;*

2. *A functor $\theta : \sigma \to \mathcal{A}$ agreeing with σ on objects (i.e. a labelling of σ with arrows of $\mathcal{A}$);*

3. *A multiset L of loops from $\mathcal{A}$.*

In Chapter 4 we will give a description of the free category $F\mathcal{A}$ in logical terms, and in Chapter 6 we describe a generalisation where the generators $\mathcal{A}$ form a polycategory rather than a category. In the next chapter we turn our attention to the connection between compact closed categories and quantum mechanics.

Chapter 3

Categorical Quantum Mechanics

We now reformulate the Hilbert space presentation of quantum mechanics in the abstract setting of strongly compact closed categories. This approach was initiated by Abramsky and Coecke [AC04] and was further developed in [Coe05, Sel05, AC05, CP07]; here we follow [AC04] closely and neglect the refinements introduced in later papers. In presenting this material, we aim to give a concrete operational meaning to the abstract structures that will appear in the formal categories constructed in later chapters. A noteworthy point is that the distinction between the different kinds of concrete operation — preparation, evolution, and measurement — are captured by the *types* of their abstract counterparts.

Throughout this thesis we will consider only what might be termed *multiplicative* aspects of quantum mechanics[1]; we do not consider the non-deterministic and probabilistic aspects of the theory. Importantly it is still possible to discuss the *action* of a measurement in such a setting, if not the range of its possible outcomes. To make this statement precise, the notion of *run* must be introduced.

Recall that a quantum measurement on a given state produces a probabilistic state change; hence the execution of a quantum process containing measurements can be described as a tree of possible behaviours, where each path is weighted by a probability. A *run* is a single path in this tree, together with its associated probability. From an extensional point of view, a run is simply an un-normalised state; by considering all the possible runs of the process a probability distribution over its output states can be obtained. In the following, however, the branching behaviour will be ignored and runs will be considered in isolation. Note that, as well as uniquely defines a state, a run bears additional quantitative information in the form of the weight. While it is nonsensical to consider this a probability in the absence of other possible runs, weights do give rise to probabilities when combined with the branching behaviour.

The first three sections of this chapter describe multiplicative quantum mechanics in the concrete setting, and then present its abstract formulation using the language of strongly compact closed categories. The final section discusses the construction of free models of abstract quantum mechanics from free com-

[1] Another possible term would be *post-selected*.

pact closed categories.

3.1 Multiplicative Quantum Mechanics

In the absence of measurements, quantum mechanics is a deterministic theory. The evolution of a closed quantum system is determined exactly by the time-dependent Schrödinger equation: if the present state is known any future state can be calculated from the solution of the equation. Indeed the evolution map is unitary, and hence invertible, so the present determines the past as well as the future. If one considers the universe to be a quantum system then it is surely closed and there is no need to consider quantum indeterminism or probabilities at all.

There are of course grave practical, as well as philosophical, objections to such an approach. Lacking knowledge of the "wave-function of the universe", it is necessary to consider the interaction of quantum systems with macroscopic agents governed by classical physics, and here non-determinism enters the picture.

Classical agents interact with quantum systems via measurements. For a given quantum system with a discrete observable, the outcome of the measurement — i.e. what the observer will see — is governed by a probability distribution, dependent on both the measurement performed and the state of the system. Quantum measurement has a side-effect: the state vector is projected onto the eigenspace determined by the measurement outcome. The evolution of a system subject to measurement can be seen as a branching tree in which each path is deterministic, and the choice of path is dictated by the probabilistic measurement outcomes. By conditioning over these outcomes we can fix a particular deterministic evolution for the system.

The physical meaning of such a branch is unambiguous: we consider a repeated experiment containing measurements, and when analysing its behaviour we simply discard all those runs of the experiment where the measurements do not agree with the chosen path. If we view our experiment as a quantum computer there are clear complexity implications for such a view. Aaronson [Aar05] considers a complexity class of quantum computations with post-selection and finds that it is equivalent to the class **PP** of classical polynomial time probabilistic computations. This class contains **NP**, hence it is outside of what we might consider computationally feasible.

We do not consider complexity issues here. We study these deterministic paths because our aim is understand the state transformations which are possible for a quantum computer. Expressed another way, we are interested in purely quantum computation without considering any classical information or control that may be present. Since the state transformations induced by measurements are fundamental to several approaches to quantum computation [RB01, GC99] we must incorporate *measurement actions* into our presentation in order to discuss these models. Fortunately, this is not problematic.

We restate the postulates of quantum mechanics, but this time omit all reference to probabilistic branching.

1. To each quantum system we associate a finite dimensional Hilbert space H, its *state space*. A composite quantum system whose subsystems have state spaces $H_1, \ldots, H_n$ has the state space $H_1 \otimes \cdots \otimes H_n$.

2. The system's state is represented by a *ray* $|\psi\rangle = \{c\psi \in H \mid c \in \mathbb{C}\}$ for some vector ψ in the state space H.
3. The evolution of a closed system is described by a *unitary* map for each discrete time step $|\psi_1\rangle \xrightarrow{U} |\psi_2\rangle$.
4. The action of a measurement is described by a *projector* $P : H \to H$ such that P is self-adjoint and idempotent up to a scalar factor.

Since the set of vectors of H is isomorphic to the set of linear maps $\mathbb{C} \to H$, there is no danger in identifying the state $|\psi\rangle$ and the corresponding ray in $\mathbf{FDHilb}(\mathbb{C}, H)$. Viewing states as maps rather than elements of a set makes the translation into categorical language simpler. We will usually identify a state of a system and the process which produced it, and so will use the terms "state" and "preparation" interchangeably depending on which emphasis is desired.

In addition to the four axioms above, we make one further restriction. We consider only measurements which are *non-degenerate*: every projector P has a 1-dimensional image. In this case, there exists a vector ϕ such that P factors into components,

$$P = H \xrightarrow{\langle\phi|} \mathbb{C} \xrightarrow{z} \mathbb{C} \xrightarrow{|\phi\rangle} H.$$

Hence each non-destructive measurement can be viewed as a destructive projection followed immediately by the preparation of a fresh state $|\phi\rangle$. Note that the scalar factor z may be equally well absorbed into either part. Since the state preparation is already accounted for by axiom 2, we simplify axiom 4 as

4' The action of a measurement is described by a ray

$$\langle\psi| = \{c\langle\psi \mid \cdot\rangle \in (H \to \mathbb{C}) \mid c \in \mathbb{C}\}$$

for some vector $\psi \in H$.

Hence measurement actions may be viewed as dual to states.

Remark. A subtlety worth pointing out early is that the identification of states and preparations is valid only under the hypothesis that all other paths in the computation may be neglected. Otherwise the scalar weights on a path correspond to the probability (or rather, the amplitude) of that path's success. Hence we will pay close attention to the scalar factors, and *not* identify arrows which differ only by a scalar. In other words, we work with runs rather than states.

3.2 FDHilb as a strong compact closed category

Theorem 3.1. *The category of finite dimensional Hilbert spaces and linear maps,* ***FDHilb****, is strongly compact closed with the following structure:*

- $(\boldsymbol{FDHilb}, \otimes, \mathbb{C})$ *is symmetric monoidal with* $\otimes$ *the usual tensor product of Hilbert spaces;*
- *given a Hilbert space* H*, define* H^* *to be the space containing the same set of vectors as* H *with the following scalar multiplication and inner product*

$$\lambda \bullet_* v := \overline{\lambda} \bullet v, \qquad \langle v \mid u\rangle_* := \langle u \mid v\rangle,$$

and addition as in H*;*

- *the functor $(\cdot)^\dagger$ is defined as the identity on objects and, for each linear map $f : H_1 \to H_2$, we define $f^\dagger$ as the unique map satisfying*

$$\langle v \mid fu \rangle = \langle f^\dagger v \mid u \rangle.$$

- *Let $\{e_i\}_i$ be a basis for H; define $\eta_H : I \to H^* \otimes H$ by*

$$\eta_H : 1 \mapsto \sum_i e_i \otimes e_i.$$

Proof. I take for granted that **FDHilb** is a strict symmetric monoidal category. The assignment $H \mapsto H^*$ is trivially involutive, so we must show that it is monoidal. The identity map for $H_1 \otimes H_2$ gives a linear isomorphism $H_1^* \otimes H_2^* \to (H_1 \otimes H_2)^*$. Since the addition is the same in both, it remains to check that the scalar multiplication is preserved. Suppose that $\psi \otimes \phi$ is a vector in $H_1^* \otimes H_2^*$; then

$$\lambda \bullet_{1^*2^*} (\phi \otimes \psi) = (\lambda \bullet_{1^*} \phi) \otimes \psi = (\overline{\lambda} \bullet_1 \phi) \otimes \psi = \overline{\lambda} \bullet_{12} (\phi \otimes \psi) = \lambda \bullet_{(12)^*} (\phi \otimes \psi).$$

Similarly, the assignment $z \to \overline{z}$ is linear $\mathbb{C} \to \mathbb{C}^*$:

$$z \bullet 1 = z \mapsto \overline{z} = \overline{z} \bullet 1 = z \bullet_* 1$$

hence $(\cdot)^*$ is (strongly) monoidal.

To see that $(\cdot)^\dagger$ is a strict monoidal functor, suppose that we have Hilbert spaces and linear maps

$$A \xrightarrow{f} B \xrightarrow{g} C;$$

then $\langle (gf)^\dagger c \mid a \rangle = \langle c \mid gfa \rangle = \langle f^\dagger g^\dagger c \mid a \rangle$ so $(fg)^\dagger = g^\dagger f^\dagger$, and also $\langle a \mid 1_A a \rangle = \langle 1_A a \mid a \rangle$ so $1_A^\dagger = 1_{A^\dagger} = 1_A$, hence $(\cdot)^\dagger$ is a functor. Since **FDHilb** is strict and $(\cdot)^\dagger$ is the identity on objects the natural maps α, ρ, and λ are all identities; this fact establishes

$$\alpha^{-1} = \alpha^\dagger, \qquad \rho^{-1} = \rho^\dagger, \qquad \lambda^{-1} = \lambda^\dagger.$$

In addition,

$$\langle b \otimes a \mid \sigma_{A,B}(a \otimes b) \rangle = \langle \sigma_{A,B}^{-1}(b \otimes a) \mid a \otimes b \rangle$$

hence $\sigma^\dagger = \sigma^{-1}$ as required. To complete the proof that $(\cdot)^\dagger$ is a strict monoidal functor, we have

$$\begin{aligned} \langle (f \otimes g)^\dagger (b \otimes c) \mid a \otimes b' \rangle &= \langle b \otimes c \mid (f \otimes g)(a \otimes b') \rangle \\ &= \langle b \otimes c \mid (fa \otimes gb') \rangle \\ &= \langle b \mid fa \rangle \langle c \mid gb' \rangle \\ &= \langle f^\dagger b \mid a \rangle \langle g^\dagger c \mid b' \rangle \\ &= \langle (f^\dagger \otimes g^\dagger)(b \otimes c) \mid a \otimes b' \rangle, \end{aligned}$$

Hence $(\cdot)^\dagger \otimes (\cdot)^\dagger = (\cdot \otimes \cdot)^\dagger$ as required.

Finally suppose that $\psi, \phi \in A$; then

$$\begin{aligned}\langle \psi \otimes \phi \mid \eta_A(1) \rangle &= \langle \psi \otimes \phi \mid \sum_i a_i \otimes a_i \rangle \\ &= \sum_i \langle \psi \mid a_i \rangle \langle \phi \mid a_i \rangle \\ &= \langle \phi \mid \psi \rangle.\end{aligned}$$

Hence ϵ_A is the linear map generated by $\psi \otimes \phi \mapsto \langle \psi \mid \phi \rangle$. From which

$$\psi \xmapsto{1_A \otimes \eta_A} \psi \otimes (\sum_i a_i \otimes a_i) = \sum_i \psi \otimes a_i \otimes a_i \xmapsto{\epsilon_A \otimes 1_A} \sum_i \langle \psi \mid a_i \rangle a_i = \psi$$

satisfying the required commuting triangle to make **FDHilb** compact closed. □

Remark. In the above, H^* is not the usual dual space — that is, the space of linear maps $H \to \mathbb{C}$. Taking it as such would give a natural isomorphism between H^{**} and H, but not a true involution.

3.3 Categorical Quantum Mechanics

The features of Hilbert space used in the postulates of quantum mechanics without branching can all be expressed purely in terms of the strongly compact closed structure of **FDHilb**. By rephrasing the postulates in categorical language, we create an abstract version of quantum mechanics which may be understood in any strongly compact closed category.

Given a strongly compact closed category $\mathcal{C}$, the postulates of quantum mechanics over $\mathcal{C}$ are as follows.

1. To each quantum system we associate an object A of $\mathcal{C}$, its state space. The system's state is represented by a ray; a subset S_ψ of $\mathcal{C}(I, A)$ determined by a point $|\psi\rangle : I \to A$, such that $S_\psi = \{s \bullet |\psi\rangle \mid s \in \mathcal{C}(I, I)\}$.

2. Each discrete step of the evolution of a closed system is represented by a unitary map $f : A \to B$.

3. The action of a measurement upon a system is represented by a projector $\langle \psi | : A \to I$.

4. A composite system whose subsystems have state spaces $A_1, \ldots, A_n$ is has state space $A_1 \otimes \cdots A_n$.

The scalar multiplication in postulate 1 has two effects: it normalises all the states and it cancels out global phase. We might think of splitting the idea of state into two. Firstly as *preparation*, which is simply a point $|\psi\rangle : I \to A$; and, secondly, as a *run* which consists of any arrow in the category times a scalar factor, considered to be its amplitude. Hence each member of the set S_ψ can be viewed as the preparation $|\psi\rangle$ with some amplitude.

A point of note is the relation between the unit maps and the Bell state. Taking Q as the space of qubits, the map $\eta_Q : I \to Q^* \otimes Q$ sends 1 to the state $|0^*0\rangle + |1^*1\rangle$; dually, the counit projects this vector onto 1 and sends the rest of

the space to 0. That this pair of preparation and measurement action performs the teleportation protocol is essentially the statement of the defining axiom of a compact closed category. Hence there is a notion of Bell state, Bell projection and teleportation at every object in every compact closed category.

3.4 Free Models

In the subsequent chapters we will define models of the categorical formulation of quantum mechanics which are *freely constructed*, that is, generated from a category of basic elements by iterating the algebraic operations which define the desired structure. In fact, we will spend the bulk of the next few chapters defining free models of the compact closed structure only. In this section we explain how this seeming gap, between categorical quantum mechanics and bare compact closed categories, is bridged.

There are several reasons why free models are of interest. Foremost among these is the fact that a free model contains only those equations which follow directly from its algebraic structure. Hence we can distinguish properties depending upon whether they rely purely upon the compact closed structure or equation which hold between the generators, or indeed the interaction of both. There is no chance to misled by contingent equalities of any particular matrix representation.

This point of view is informed by the intended application to quantum computing. A programming language will provide some small number of basic operations at ground types from which all programs can be constructed by using the various methods of combination and consumption provided by the language. Realistic hardware — or experimental equipment — will implement a similarly limited set of low-level transformations. This is no limitation on the expressiveness of the formalism, since various sets of quantum gates have been shown to be universal, or approximately so ([NC00] Chapter 4.5 has a summary). This intuition suggests that freely constructed models of our axioms will provide the most natural situation to develop abstract quantum mechanics.

There is a second, more technical reason for restricting attention to freely constructed categories. In any category with equational structure we should like to know which equations hold by virtue of that structure. Such coherence questions are effectively word problems over the objects and arrows of the category. The ideal case resembles Mac Lane's celebrated coherence theorem [ML63, ML97], which gives a description of the equations which must hold in any monoidal category exclusively in terms of the natural transformations which make up the monoidal structure, and without reference to the other arrows which may be present in the category. The remarks at the end of [KL80] point out that the equations which must hold in a compact closed setting cannot be so neatly described. Kelly shows in a sequence of papers, but principally [Kel92], that such an absolute description is possible only when the structure arises from a "club", and indeed that compact closed categories do not so arise. Hence the best possible account of the structure of a compact closed category is the explicit description of the free construction relative to some generating category.

So much for compact closed categories. We are concerned here with *strongly* compact closed categories. It will turn out that the additional structure provided

by the involution $(\cdot)^\dagger$ can be largely neglected.

Selinger [Sel05] defines a *dagger category* as any category equipped with an identity-on-objects, involutive, contravariant functor; elsewhere these have been called *involutive categories* [Abr05, AD06]. A strongly compact closed category[2] can then be defined as a dagger category which is compact closed and additionally satisfies:

$$(f \otimes g)^\dagger = f^\dagger \otimes g^\dagger \,, \tag{3.1}$$

$$\alpha^\dagger_{A,B,C} = \alpha^{-1}_{A,B,C} \,, \tag{3.2}$$

$$\lambda^\dagger_A = \lambda^{-1}_A \,, \tag{3.3}$$

$$\sigma^\dagger_{A,B} = \sigma^{-1}_{A,B} \,, \tag{3.4}$$

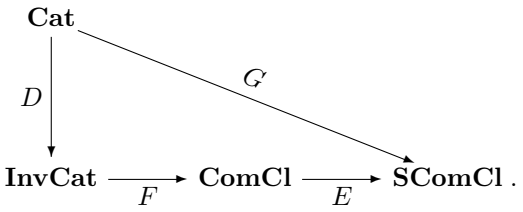
$$\epsilon^\dagger_A = \sigma_{A^*,A} \circ \eta_A \,. \tag{3.5}$$

In other words, a strongly compact closed category is a compact closed dagger category where $(\cdot)^\dagger$ preserves the monoidal and compact closed structure exactly. This suggests that it is possible to construct the free strongly compact closed category on $\mathcal{A}$ in three steps, corresponding to the dagger, the compact closed structure, and the coherence of the two.

Consider the three functors,

$$\begin{aligned} D &: \mathbf{Cat} \longrightarrow \mathbf{InvCat} \,, \\ F &: \mathbf{Cat} \longrightarrow \mathbf{ComCl} \,, \\ G &: \mathbf{Cat} \longrightarrow \mathbf{SComCl} \,, \end{aligned}$$

which respectively send a category to the free dagger category, free compact closed category, and free strongly compact closed category generated by it. Given some category $\mathcal{A}$, $FD\mathcal{A}$ is not strongly compact closed since the functor $(\cdot)^\dagger$ is not defined on every arrow. However, as shown in [Abr05], it can be extended by taking Equations (3.1)–(3.5) to define $(\cdot)^\dagger$ on the arrows where it was not already defined. Calling this (trivial) embedding functor E, we now have:



It is easy to see that $EFD\mathcal{A}$ is indeed the free strongly compact closed category on $\mathcal{A}$ since it is freely compact closed and includes an arrow $f^\dagger : B \to A$ for each $f : A \to B$ such that $f^\dagger$ is formally distinct from all other arrows except where the coherence laws require an equation.

To combine the functors in the other order, let **InvComClCat** be the category of compact closed categories with involutions, and let D' be the restriction

[2]Selinger uses the term "dagger compact closed" for strongly compact closed. The fact that a dagger compact closed category is different to a compact closed dagger category seems sufficient reason to avoid this terminology.

of D to compact closed categories, i.e. the functor making

$$\begin{array}{ccc} \mathbf{ComCl} & \xrightarrow{D'} & \mathbf{InvComClCat} \\ \downarrow{\scriptstyle U} & & \downarrow{\scriptstyle U'} \\ \mathbf{Cat} & \xrightarrow[F^\dagger]{} & \mathbf{InvCat} \end{array}$$

commute, with U, U' the evident forgetful functors. Then $D'F\mathcal{A}$ is dagger category by its construction, and it is also compact closed since every object inherits a dual from $F\mathcal{A}$. It is not strongly compact closed since $f^\dagger$ is always distinct from f on non-identity arrows, and thus Equations (3.1)–(3.5) do not hold.

Let $\mathcal{B} = D'F\mathcal{A}$ and let R_{AB} be the least equivalence relation on the hom-set $\mathcal{B}(A, B)$ such that

- $(f, g) \in R_{A,B}$ if $f = g$ by Equations (3.1)–(3.5)
- $R(A, B) = R_{A,B}$ defines a functor $\mathcal{B}^{\mathrm{op}} \times \mathcal{B} \to \mathbf{Set}$;
- $R(A, B) \times R(C, D) \subseteq R(A \otimes C, B \otimes D)$

Since R is a functor we have

$$R(A, B) \cong R(A \otimes I, B) \cong R(I \otimes A, B) \cong R(A, I \otimes B) \cong R(A, B \otimes I)$$

via the left and right monoid unit morphisms. Since it respects tensor we also have $R(A, B) \cong R(B^*, A^*)$. Therefore the quotient functor $Q : \mathbf{InvComClCat} \to \mathbf{SComCl}$ generated by R is a compact closed functor. We then have:

$$\begin{array}{ccccc} \mathbf{Cat} & \xrightarrow{F} & \mathbf{ComCl} & \xrightarrow{D'} & \mathbf{InvComClCat} \\ \downarrow{\scriptstyle D} & \searrow{\scriptstyle G} & & & \downarrow{\scriptstyle Q} \\ \mathbf{InvCat} & \xrightarrow[F]{} & \mathbf{ComCl} & \xrightarrow[E]{} & \mathbf{SComCl} \end{array}$$

Hence in a free strongly compact closed category the role of the $(\cdot)^\dagger$ is restricted to the generating category $\mathcal{A}$ and may be otherwise neglected. By way of emphasis, consider the case where $\mathcal{A} = \mathbf{1}$, the category with one object and no non-identity arrows. We have $D\mathbf{1} = \mathbf{1}$, and hence $G\mathbf{1} = F\mathbf{1}$.

The construction of the free strong compact closed structure provides a number of unitary and self-adjoint maps with which to carry out abstract quantum mechanics. The monoidal natural isomorphisms are all unitary. For each A, the non-destructive measurement action,

$$A \otimes A^* \xrightarrow{\epsilon_A} I \xrightarrow{\eta_A} A^* \otimes A \xrightarrow{\sigma_{A^*,A}} A \otimes A^*$$

is self-adjoint, upto a scalar factor of $\dim_A$. These arrows provide limited scope for defining interesting algorithms. In light of the discussion above, any additional unitaries must come from from the generating category $\mathcal{A}$. Rather than simply considering the free dagger category $D\mathcal{A}$, it would be more profitable to consider a quotient of $D\mathcal{A}$ by some set of equations. For example:

- $f^\dagger f = 1_A$ and $ff^\dagger = 1_B$ for all $f : A \to B$, making $\mathcal{A}$ into a unitary groupoid;
- $f^\dagger = f$ for some chosen f, since many common quantum logic gates are defined by self-adjoint maps;
- more generally, commutation relations $fg = hf$ for some arrows f, g, h, may be required, for example to represent a symmetry group.

The procedure is rather similar to that above. Let $\mathcal{E}$ be a set of equations on the arrows of $\mathcal{C} = D\mathcal{A}$ and let $R_{A,B}$ be the least equivalence relation on $\mathcal{C}(A, B)$ such that

- $(f, g) \in R_{A,B}$ if $f = g$ is an equation of $\mathcal{E}$;
- $R(A, B) = R_{A,B}$ defines a functor $\mathcal{C}^{\mathrm{op}} \times \mathcal{C} \to \mathbf{Set}$.

Given such a functor R, there exists a universal arrow Q to the unique quotient category $\mathcal{C}/R$; we must construct a functor $S : F\mathcal{C}^{\mathrm{op}} \times F\mathcal{C} \to \mathbf{Set}$ such that the quotients commute with the free construction of the compact closed structure as shown below.

$$\begin{array}{ccc} \mathcal{C} & \xrightarrow{Q_R} & \mathcal{C}/R \\ {\scriptstyle \Psi_{\mathcal{C}}}\downarrow & & \downarrow{\scriptstyle \Psi_{\mathcal{C}/R}} \\ F\mathcal{C} & \xrightarrow[Q_S]{} & (F\mathcal{C})/S = F(\mathcal{C}/R) \end{array}$$

It suffices to lift the relations R_{AB} to tensors of objects. Let $S_{X,Y}$ be the least equivalence on $F\mathcal{C}(X, Y)$ such that

- $S_{A,B} = R_{A,B}$ whenever A and B are objects of $\mathcal{C}$;
- $S(A, B) = S_{A,B}$ defines a functor $F\mathcal{C}^{\mathrm{op}} \times F\mathcal{C} \to \mathbf{Set}$.
- $S(A, B) \times S(C \otimes D) \subseteq S(A \otimes C, B \otimes D)$.

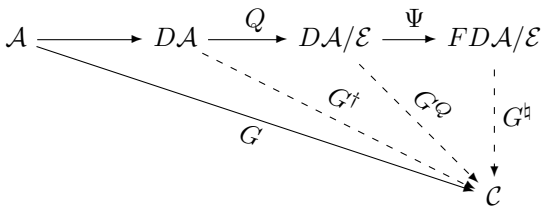
This relation defines a quotient functor $Q_S : F\mathcal{C} \to (F\mathcal{C})/S$, which preserves the compact closed structure exactly. Since this structure is preserved, the characterisation of Theorem 2.8 still applies. Suppose then that $f = (\sigma, \theta, L)$, $g = (\sigma', \theta', L')$ and $Qf = Qg$. Since Q_S doesn't change the objects we must have $\sigma = \sigma'$, which implies that θ and θ' may differ only on the arrows of the involution. Since each of these defines an arrow in $\mathcal{C}$, Q_S will only equalise them when Q_R does, hence $\theta : \sigma \to \mathcal{C}/R$. The same applies to the loops. Therefore $(F\mathcal{C})/S$ is contained in $F(\mathcal{C}/R)$. The converse is similar. Therefore, it is possible to characterise the free compact closed structure without reference to any equations imposed upon $D\mathcal{A}$, and perform the quotient upon $FD\mathcal{C}$ as needed.

With these thoughts in mind, we now present a recipe for the construction of free models of categorical quantum mechanics.

- Start with a category $\mathcal{A}$; this provides the ground types and primitive operations.

- For every arrow $f : A \to B$ augment $\mathcal{A}$ with a formally distinct arrow $f^\dagger : B \to A$, such that $1_A^\dagger = 1_A$ and $(f \circ g)^\dagger = g^\dagger \circ f^\dagger$ to construct the free dagger category $D\mathcal{A}$.
- Quotient $D\mathcal{A}$ by some set of equations $\mathcal{E}$.
- Construct the free compact closed category $F(D\mathcal{A}/\mathcal{E})$.

These steps ensure that the resulting category has the required elements to carry out quantum mechanics in the abstract setting.



Since each of the horizontal arrows in the digram above is universal, if there is a functor $G : \mathcal{A} \to \mathcal{C}$ to any strongly compact closed category, such that G preserves the equations $\mathcal{E}$ then there exists a unique compact closed functor from the freely constructed model $FD\mathcal{A}/R$ such that G factors through it. Hence, in order to interpret the free model in a concrete category — typically **FDHilb**— it suffices to provide a functor interpreting the primitives.

In the following chapters we shall set about providing an explicit description of the freely constructed compact closed part of this structure, without any undue worry about the other aspects. In keeping with this, we shall henceforth drop the rather heavy notation $FD\mathcal{A}/R$, and speak simply of the free compact closed category $F\mathcal{A}$ on a category of generators $\mathcal{A}$.

Chapter 4

Multiplicative Categorical Quantum Logic

This chapter introduces **mCQL** , a logic which permits formal reasoning within the strongly compact closed framework described in the previous chapter. As such, it will be our first pass at a formal syntax for abstract quantum mechanics.

The syntax of **mCQL** is constructed relative to a class of basic types and operations upon which an abstract quantum theory is built by imposing connectives and structural rules which carry the compact closure. To be more precise, we begin with a base category $\mathcal{A}$ and construct the free compact closed category $F\mathcal{A}$ upon it.

$$\mathbf{Cat} \underset{U}{\overset{F}{\underset{\longleftarrow}{\overset{\longrightarrow}{\bot}}}} \mathbf{Com}$$

The constructed category $F\mathcal{A}$ forms the setting for our abstract quantum mechanics. The syntax of **mCQL** is a formal representation of this structure: the formulae and proofs encode respectively the objects and arrows of $F\mathcal{A}$, which we may think of physically as abstract state spaces and quantum processes between them.

Before continuing, it is worth disposing of an "impedance mismatch". This chapter is concerned with developing a theory for the free *compact closed* structure, whereas previously we claimed that the requisite framework for this part of quantum mechanics is a *strongly* compact closed category. The essential extra structure of the strongly compact closed setting is the involution $(\cdot)^\dagger$, and as described in Chapter 3, if $\mathcal{A}$ has a suitable involution we can lift it to $F\mathcal{A}$ to produce a strongly compact closed category. The action of $(\cdot)^\dagger$ on the compact closed structural maps is either trivial or does not introduce any new maps, merely sending units to counits, and so on. Hence the effect of the involution is effectively restricted to $\mathcal{A}$. Therefore we will assume that $\mathcal{A}$ is already freely augmented with a suitable involution and otherwise neglect the "strong" part of strongly compact closed.

In what follows, we will offer two presentations of **mCQL**, firstly as a sequent calculus, and secondly as a system of proof-nets. The sequent calculus has the advantage that the relationship between the syntax and its interpretation is

clearer; however the proof-nets presentation stands in much closer relation to the model. Indeed proof-nets for **mCQL** will give a faithful and fully complete representation of $F\mathcal{A}$.

4.1 Formulae

In the following, $\mathcal{A}$ is the chosen base category, also known as the category of axioms, or category of atoms.

Definition 4.1. The *formulae* of **mCQL** are built from the following grammar:

$$F ::= A \mid A^* \mid F \otimes F,$$

where A ranges over the objects of $\mathcal{A}$, which we shall refer to as *atoms*. We define $(\cdot)^*$ on arbitrary formulae by the following equations:

$$\begin{aligned} X^{**} &= X \\ (X \otimes Y)^* &= Y^* \otimes X^* \end{aligned}$$

We use the notational convention that upper case letters A, B, C from the start of the Latin alphabet are atoms and those from the end of the alphabet X, Y, Z are arbitrary formulae. Upper case Greek letters Γ, Δ, Σ signify lists of formulae.

We shall use *axiom* synonymously with *arrow of* $\mathcal{A}$.

4.2 Sequent Calculus

Definition 4.2. An **mCQL** *sequent* is of the form

$$\Gamma \vdash \Delta; [L]$$

where Γ, Δ are lists of formulae and L is a multiset of loops.

The incorporation of the loops in the definition of sequent is slightly misleading: the loop sets are a proof decoration rather than a property of the sequents themselves. The loops are syntactic representatives for the scalars $I \to I$; they may be considered weights or amplitudes of the particular deterministic process represented by the proof. This quantitative information should not form part of the type, so two sequents are defined to be equal if they differ only by loops.

Further, the inclusion of loops in the syntax allows a minor technical improvement on [Shi96] in that troublesome cuts can be eliminated by introducing a loop, and hence there is no need for an auxiliary notion of normal form.

Warning!. It is necessary to distinguish between occurrences of formulae in sequents and throughout proofs, otherwise one can arrive at situations where it is impossible to see what is going on. In the following it is tacitly assumed that all formulae occur uniquely. This defect is remedied in the proof-net presentation of the next section.

Definition 4.3. An **mCQL** proof is a tree of sequents joined by the inference rules shown in figure 4.1. The inferences at the leaves of the tree must be either f-axioms or h-units.

Structure Group: σ, τ permutations.

$$\frac{\Gamma, X \vdash X, \Delta \,;[L]}{\Gamma \vdash \Delta \,;[L]}\,(\text{cut}) \qquad \frac{\Gamma \vdash \Delta \,;[L] \qquad \Gamma' \vdash \Delta' \,;[L']}{\Gamma, \Gamma' \vdash \Delta, \Delta' \,;[L, L']}\,(\text{mix})$$

$$\frac{\Gamma \vdash \Delta \,;[L]}{\tau(\Gamma) \vdash \sigma(\Delta) \,;[L]}\,(\text{exch})$$

Multiplicative Group:

$$\frac{\Gamma, X, Y \vdash \Delta \,;[L]}{\Gamma, X \otimes Y \vdash \Delta \,;[L]}\,(\text{L}\otimes) \qquad \frac{\Gamma \vdash X, Y, \Delta \,;[L]}{\Gamma \vdash X \otimes Y, \Delta \,;[L]}\,(\text{R}\otimes)$$

$$\frac{\Gamma \vdash \Delta, X \,;[L]}{\Gamma, X^* \vdash \Delta \,;[L]}\,(\text{L}*) \qquad \frac{X, \Gamma \vdash \Delta \,;[L]}{\Gamma \vdash X^*, \Delta \,;[L]}\,(\text{R}*)$$

$\mathcal{A}$-Group: where $A \xrightarrow{f} B \xrightarrow{g} A$ are axioms, and h is a loop.

$$\frac{}{A \vdash B \,;[]}\,(f\text{-axiom}) \qquad \frac{\Gamma, A \vdash B, \Delta \,;[L]}{\Gamma \vdash \Delta \,;[L]}\,(g\text{-cut})$$

$$\frac{}{\vdash \,;[h]}\,(h\text{-unit})$$

Figure 4.1: Inference Rules for **mCQL**

Note that since $\mathcal{A}$ is a category there is an 1_A-axiom rule for every object A of $\mathcal{A}$, but, for reasons of technical convenience, we only include identity axioms for atomic formulae. Identity axioms are admissible for all formulae, as a consequence of the full completeness theorem, but we will not need them. We *do* however include cut rules at all formulae, in addition to those parameterised by the arrows of $\mathcal{A}$.

Example 4.4. Suppose we have axioms $B \xrightarrow{f} A$ and $C \xrightarrow{g} D \xrightarrow{h} B$.

$$\dfrac{\dfrac{\dfrac{\dfrac{}{B \vdash A;[]}\,(f\text{-axiom}) \qquad \dfrac{}{A \vdash A;[]}\,(1_A\text{-axiom})}{B, A \vdash A, A;[]}\,(\text{mix})}{B, A \vdash A \otimes A \,;[]}\,(\text{R}\otimes) \qquad \dfrac{}{C \vdash D;[]}\,(g\text{-axiom})}{B, A, C \vdash A \otimes A, D;[]}\,(\text{mix})}{A, C \vdash A \otimes A;[]}\,(h\text{-cut})$$

The cut rule, as shown here, might be better described as a trace rule, and in the semantics for **mCQL** proofs it is indeed interpreted by the trace. By proposition 2.4 of [AHS02] we have

$$g \circ f = \mathrm{Tr}^B_{A,C}(\sigma_{B,C} \circ (f \otimes g))$$

Structure Group: σ, τ permutations.

$$\frac{f : \Gamma \otimes A \to A \otimes \Delta}{\mathrm{Tr}^{A}_{\Gamma,\Delta}(\sigma \circ f) : \Gamma \to \Delta}\,(\text{cut}) \qquad \frac{f : \Gamma \to \Delta \qquad g : \Gamma' \to \Delta'}{(f \otimes g) : \Gamma \otimes \Gamma' \to \Delta \otimes \Delta'}\,(\text{mix})$$

$$\frac{f : \Gamma \to \Delta}{\sigma \circ f \circ \tau^{-1} : \tau(\Gamma) \to \sigma(\Delta)}\,(\text{exch})$$

Multiplicative Group:

$$\frac{f : \Gamma \to \Delta \otimes X}{(1_\Delta \otimes \epsilon_X) \circ (f \otimes 1_{X^*}) : \Gamma \otimes X^* \to \Delta}\,(\text{L}*)$$

$$\frac{f : X \otimes \Gamma \to \Delta}{(f \otimes 1_{X^*}) \circ (1_\Gamma \otimes \eta_X) : \Gamma \to X^* \otimes \Delta \,;[L]}\,(\text{R}*)$$

No interpretation of R$\otimes$ or L$\otimes$

$\mathcal{A}$**-Group**: where $A \xrightarrow{f} B \xrightarrow{g} A$ are axioms, and h is a loop.

$$\frac{}{f : A \to B}\,(f\text{-axiom}) \qquad \frac{f : \Gamma \otimes A \to B \otimes \Delta}{\mathrm{Tr}^{A \otimes B}_{\Gamma,\Delta}(\sigma \circ (f \otimes g)) : \Gamma \to \Delta}\,(g\text{-cut})$$

$$\frac{}{\epsilon_A \circ \ulcorner h \urcorner : I \to I}\,(h\text{-unit})$$

Figure 4.2: Semantics for rules of **mCQL**

for $f : A \to B$, $g : B \to C$. Hence the rule gives the usual idea of partial composition. The traditional cut rule,

$$\frac{\Gamma \vdash \Delta, A \qquad A, \Gamma' \vdash \Delta'}{\Gamma, \Gamma' \vdash \Delta, \Delta'}$$

is admissible, and we will write it as short hand for its **mCQL** derivation, viz.

$$\frac{\dfrac{\Gamma \vdash \Delta, A \qquad A, \Gamma' \vdash \Delta'}{\Gamma, A, \Gamma' \vdash \Delta, A, \Delta'}\,(\text{mix})}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\,(\text{cut})$$

Definition 4.5. To each **mCQL** proof π of $\Gamma \vdash \Delta$ we assign an arrow $[\![\pi]\!] : \bigotimes \Gamma \to \bigotimes \Delta$. This arrow is constructed from π via the correspondence between the inference rules of Figure 4.1 and the constructions on arrows shown in Figure 4.2.

While this sequent calculus encodes the categorical structure very naturally, the large number of rules makes it rather cumbersome. By taking advantage of De Morgan duality we can convert all sequents and proofs into a one-sided

$$\frac{}{\vdash A^*, B\,;[]}\,(f\text{-axiom}) \qquad \frac{}{\ulcorner f \urcorner : I \to A^* \otimes B}\,(f\text{-axiom})$$

$$\frac{}{\vdash\,;[h]}\,(h\text{-unit}) \qquad \frac{}{\eta_{A*} \circ \ulcorner h \urcorner : I \to I}\,(h\text{-unit})$$

$$\frac{\vdash \Gamma, X, X^*\,;[L]}{\vdash \Gamma\,;[L]}\,(\text{cut}) \qquad \frac{f : I \to \Gamma \otimes X \otimes X^* \otimes \Delta}{(1_\Gamma \otimes \epsilon_X \otimes 1_\Delta) \circ f : I \to \Gamma \otimes \Delta}\,(\text{cut})$$

$$\frac{\vdash \Gamma, A^*, B, \Delta\,;[L]}{\vdash \Gamma\,;[L]}\,(g\text{-cut}) \qquad \frac{f : I \to \Gamma \otimes A \otimes B^* \otimes \Delta}{(1_\Gamma \otimes \llcorner g \lrcorner \otimes 1_\Delta) \circ f : I \to \Gamma \otimes \Delta}\,(g\text{-cut})$$

$$\frac{\vdash \Gamma\,;[L]}{\vdash \sigma(\Gamma)\,;[L]}\,(\text{exch}) \qquad \frac{f : I \to \Gamma}{\sigma \circ f : I \to \sigma(\Gamma)}\,(\text{exch})$$

$$\frac{\vdash \Gamma\,;[L] \qquad \vdash \Delta\,;[L']}{\vdash \Gamma, \Delta\,;[L, L']}\,(\text{mix}) \qquad \frac{f : I \to \Gamma \qquad g : I \to \Delta}{(f \otimes g) : I \to \Gamma \otimes \Delta}\,(\text{mix})$$

$$\frac{\vdash \Gamma, X, Y\,;[L]}{\vdash \Gamma, X \otimes Y\,;[L]}\,(\otimes) \qquad (\text{no interpretation})$$

Figure 4.3: Syntax (left) and Semantics (right) for One-Sided **mCQL**

variant, as shown in Figure 4.3, and so reduce the workload while losing nothing essential. Where disambiguation is needed we write $\mathbf{mCQL}_1$ and $\mathbf{mCQL}_2$ for the one- and two-sided presentations respectively.

If Γ is the list of formulae $X_1, \ldots, X_n$, let Γ^* be the list of formulae $X_n^*, \ldots, X_1^*$.

Definition 4.6. Given an $\mathbf{mCQL}_2$ proof π of the sequent $\Gamma \vdash \Delta\,;[L]$ we can define an $\mathbf{mCQL}_1$ proof π^* of $\vdash \Gamma^*, \Delta\,;[L]$ by a direct rule for rule translation of π. Since it is self dual, both left and right rules for the tensor are translated by the same rule in the one sided system. There is no one-sided rule for the right star rule; the left star rule is translated by the exchange:

$$\frac{\vdash \Delta^*, \Gamma, X}{\vdash X, \Delta^*, \Gamma}\,(\text{exch}).$$

Proposition 4.7. *Let π be an $\boldsymbol{mCQL}_2$ proof and π^* its $\boldsymbol{mCQL}_1$ translation. Then:*

$$[\![\pi^*]\!] = \ulcorner [\![\pi]\!] \urcorner.$$

Proof. Use induction on the structure of π. The cases for axiom, unit, mix left star, and tensor are trivial. The cut rule will be omitted since it may be dealt with using the same techniques as g-cut. Hence only the three cases remain : left star, g-cut, and exchange.

Suppose that π arises from π_1 by an application of the exchange rule. By induction hypothesis, we have $[\![\pi_1^*]\!] = \ulcorner [\![\pi_1]\!] \urcorner$. Note that $\ulcorner h \circ f \circ g \urcorner = (g^* \otimes h) \circ$

$\ulcorner f \urcorner$, and if τ is a permutation on Γ then $\tau^* = \tau^{-1}$ on Γ^*. Hence

$$\ulcorner [\![\pi]\!] \urcorner = \ulcorner \sigma \circ [\![\pi_1]\!] \circ \tau^{-1} \urcorner = (\tau \otimes \sigma) \circ \ulcorner [\![\pi_1]\!] \urcorner$$

as required.

Alternatively, suppose that π arises from π_1 by an application of the g-cut rule, where $g : B \to A$ be an axiom and Let $f : X \otimes A \to Y \otimes B$ be the denotation of π_1; this gives

$$\begin{aligned} [\![\pi]\!] &= \mathrm{Tr}^{A\otimes B}_{X,Y}(\sigma \circ (f \otimes g)) \\ &= (1_Y \otimes \epsilon_{A\otimes B}) \circ \sigma \circ (f \otimes g \otimes 1_{A^*\otimes B^*}) \circ (1_X \otimes \eta_{(AB)*}). \end{aligned}$$

By induction hypothesis $[\![\pi_1^*]\!] = \ulcorner f \urcorner = (1_{X^*\otimes A^*} \otimes f) \circ \eta_{X\otimes A}$. In figure 4.5 we have $\ulcorner [\![\pi]\!] \urcorner$ along the top edge of the diagram (starting from I) and $[\![\pi^*]\!]$ along the lower edge. The cells marked $(*)$ commute because

$$\eta_{A\otimes B} = \sigma \circ (\eta_A \otimes \eta_B),$$

or its dual. The cell marked $(**)$ commutes due to the equality

$$(\epsilon_A \otimes f) \circ (1_A \otimes \eta_A) = f \circ (\epsilon_A \otimes 1_A) \circ (1_A \otimes \eta_A) = f.$$

The other cells commute either due to functoriality and/or coherence of the tensor. Hence $[\![\pi^*]\!] = \ulcorner [\![\pi]\!] \urcorner$.

Finally suppose that the last inference of π is the left star rule applied to a proof π'. Let $f = [\![\pi']\!] : \Gamma \longrightarrow \Delta \otimes X$ then by induction we have

$$[\![\pi]\!] = (1_\Delta \otimes \epsilon_X) \circ (f \otimes 1^*_X)$$

and

$$[\![\pi^*]\!] = \sigma \circ (1_{\Gamma^*} \otimes f) \circ \eta_\Gamma \ .$$

These are equal; see Figure 4.4. The outer left path is $\ulcorner [\![\pi]\!] \urcorner$ and the outer right is $[\![\pi*]\!]$; all the isomorphisms are symmetries. Hence $[\![\pi^*]\!] = \ulcorner [\![\pi]\!] \urcorner$. □

From here on we'll work exclusively with with the one sided syntax, $\mathbf{mCQL}_1$.

Theorem 4.8 (Cut-Elimination)**.** *Every **mCQL** proof can be transformed into cut-free proof of the same sequent.*

Proof. Although standard techniques largely suffice a full proof is provided. The only novelties are the rules of the $\mathcal{A}$-group, in particular the possibility of self-cuts.

Define the *rank* of cut to be the number of atomic subformulae of its cut formula; the *cut-rank* $r(\pi)$ of a proof π is the maximum rank among its cuts, with $r(\pi) = 0$ if π is cut-free. Let the *height* of a cut be the number of inferences above[1] it in the proof tree; the *cut-height* $h(\pi)$ of the proof is the sum of the heights of its cuts. We proceed by induction, with the following hypothesis: to each proof π there exists a proof π', with the same conclusions, such that

$$r(\pi) > r(\pi') \qquad \text{or} \qquad r(\pi) = r(\pi') \text{ and } h(\pi) > h(\pi').$$

[1] Unlike most trees in computer science, "up" in this context denotes "towards the leaves".

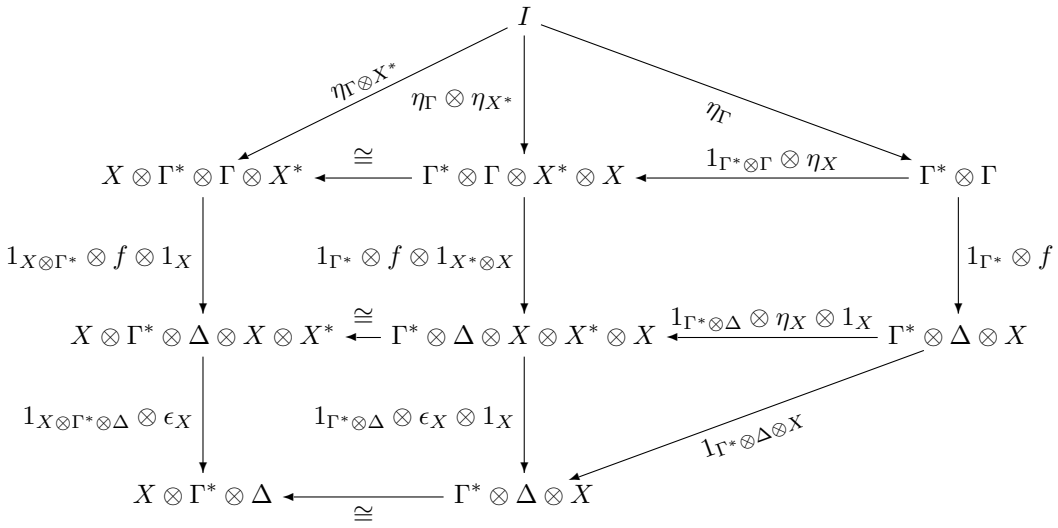


Figure 4.4: Proof of Proposition 4.7: the case for the left star rule

Assume without loss of generality that π is an **mCQL** proof containing exactly one cut, which occurs as its last inference.

Suppose $r(\pi) = 1$: the cut formulae are literals, and hence the cut is an instance of the g-cut rule, perhaps with $g = 1_A$. Suppose $g : B \to A$; then we have:

$$\dfrac{\dfrac{\vdots}{\Gamma, A^*, B\ ;[L]}\,(R)}{\Gamma\ ;[L]}\,(g\text{-cut})$$

There are three subcases depending on which rule (R) is: an axiom introducing both cut-formulae; a mix where A^* and B occur in different premises; or anything else.

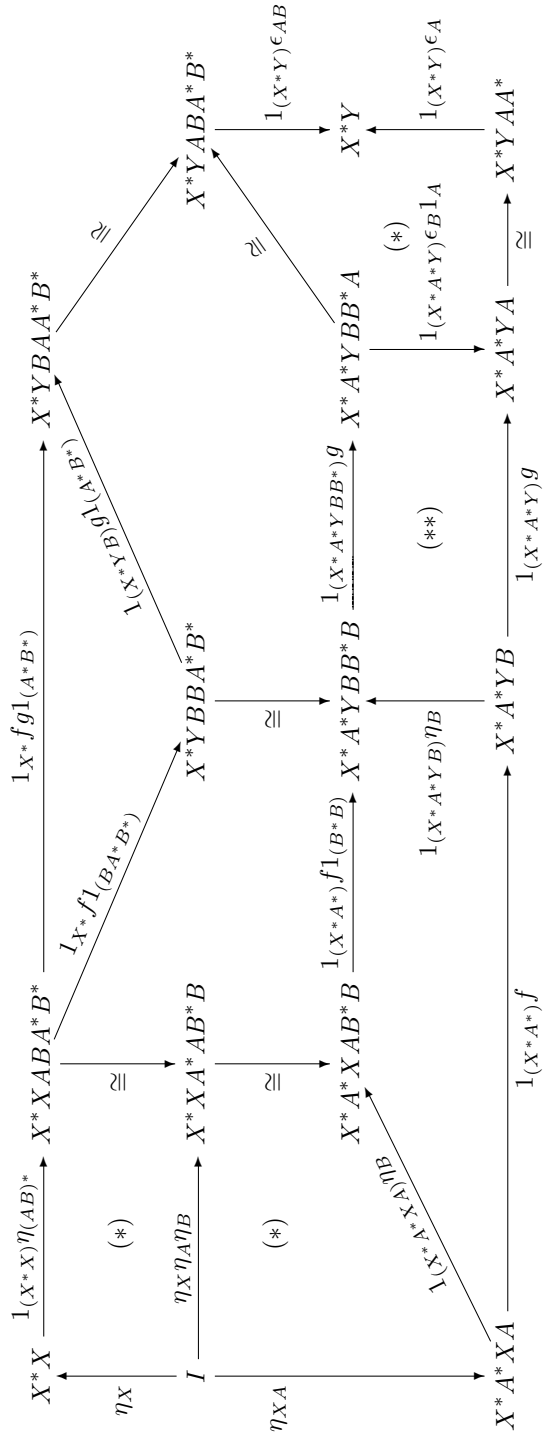
Suppose it is the first case. The proof looks like this:

$$\dfrac{\dfrac{}{\vdash A^*, B\ ;[]}\,(f\text{-axiom})}{\vdash}\,(g\text{-cut}) \tag{4.1}$$

for some axiom $f : A \to B$. This configuration represents what has been called "incestuous self-plugging" [Gir95]. In the degenerate world of compact closed categories such things are quite acceptable. The proof reduces to

$$\dfrac{}{\vdash [g \circ f]}\,((g \circ f)\text{-unit})$$

which has cut-rank zero.



NB: In this diagram, tensor is written as juxtaposition, and all isomorphisms are symmetries of the tensor.

Figure 4.5: Proposition 4.7: Case for g-cut

In the second case π has the following shape

$$\dfrac{\dfrac{\begin{array}{c}\vdots\\ \vdash \Gamma, A^* \,;[L]\end{array}}{}\,(R_1) \qquad \dfrac{\begin{array}{c}\vdots\\ \vdash B, \Delta \,;[L']\end{array}}{}\,(R_2)}{\dfrac{\vdash \Gamma, A^*, B, \Delta \,;[L, L']}{\vdash \Gamma, \Delta \,;[L, L'].}\,(g\text{-cut})}\,(\text{mix})$$

Suppose that inference R_1 is not an axiom. Then the formula A^* does not take part in (R_1) and hence the proof may be rewritten as

$$\dfrac{\dfrac{\dfrac{\begin{array}{c}\vdots\\ \vdash \Gamma', A^* \,;[L]\end{array}}{} \qquad \dfrac{\begin{array}{c}\vdots\\ \vdash B, \Delta \,;[L']\end{array}}{}\,(R_2)}{\vdash \Gamma', A^*, B, \Delta \,;[L, L']}\,(\text{mix})}{\vdash \Gamma', \Delta \,;[L, L']}\,(g\text{-cut})}{\vdash \Gamma, \Delta \,;[L, L'],}\,(R_1)$$

A similar rewrite is possible if (R_2) is not an axiom. In either case the height of the cut is reduced. If both (R_1) and (R_2) are axioms the proof is as shown below.

$$\dfrac{\dfrac{}{\vdash A^*, B \,;[]}\,(f\text{-axiom}) \qquad \dfrac{}{\vdash C^*, D \,;[]}\,(h\text{-axiom})}{\vdash A^*, D \,;[]}\,(g\text{-cut}) \tag{4.2}$$

To eliminate the cut, rewrite the proof as,

$$\dfrac{}{\vdash A^*, D \,;[].}\,((f \circ g \circ h)\text{-axiom})$$

which has rank zero.

If $r(\pi) = 1$ but neither of the cases considered above applies, then rule (R), immediately preceding the cut, has no effect on the cut-formulae. In this case it is necessarily a tensor or a mix on the side formulae, hence it is possible to rewrite the proof so that (R) comes after the cut. This will reduce the height of the cut by at least one.

The above reasoning establishes the base case of the induction; now we consider the case $r(\pi) > 1$, where a similar analysis applies.

$$\dfrac{\dfrac{\begin{array}{c}\vdots\\ \vdash \Gamma, X^* \otimes Y^*, X \otimes Y \,;[L]\end{array}}{}\,(R)}{\vdash \Gamma \,;[L]}\,(\text{cut})$$

We identify three cases: the inference (R) is an occurrence of the tensor rule which introduces one of the cut-formulae; (R) is the mix rule, and the two cut-formulae occur in different premises; or anything else. The "anything else" case is as before: since it has no effect on the premises of the cut, (R) can be pushed beneath the cut to reduce the cut height.

Otherwise, suppose that (R) is an occurrence of the mix rule as shown below.

$$\dfrac{\dfrac{\dfrac{\vdots}{\vdash \Gamma, X^* \otimes Y^* \,;[L]}\,(R_1) \qquad \dfrac{\vdots}{\vdash X \otimes Y, \Delta \,;[L']}\,(R_2)}{\vdash \Gamma, X^* \otimes Y^*, X \otimes Y, \Delta \,;[L, L']}\,(\text{mix})}{\vdash \Gamma, \Delta \,;[L, L'].}\,(\text{cut})$$

If (R_1) is not the inference introducing $X^* \otimes Y^*$ then it does not interfere with the cut-formula, hence it can be performed after the cut. Likewise for (R_2). If, on the other hand, both inferences introduce their respective cut-formulae then we have the following proof

$$\dfrac{\dfrac{\dfrac{\dfrac{\vdots}{\vdash \Gamma, X^*, Y^* \,;[L]}}{\vdash \Gamma, X^* \otimes Y^* \,;[L]}\,(\otimes) \qquad \dfrac{\dfrac{\vdots}{\vdash X, Y, \Delta \,;[L']}}{\vdash X \otimes Y, \Delta \,;[L']}\,(\otimes)}{\vdash \Gamma, X^* \otimes Y^*, X \otimes Y, \Delta \,;[L, L']}\,(\text{mix})}{\vdash \Gamma, \Delta \,;[L, L'],}\,(\text{cut}) \tag{4.3}$$

which can be rewritten to

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\vdots}{\vdash \Gamma, X^*, Y^* \,;[L]} \qquad \dfrac{\vdots}{\vdash X, Y, \Delta \,;[L']}}{\vdash \Gamma, X^*, Y^*, X, Y, \Delta \,;[L, L']}\,(\text{mix})}{\vdash \Gamma, X^*, Y^*, X, Y, \Delta \,;[L, L']}\,(\text{exch})}{\vdash \Gamma, X^*, X, Y^*, Y, \Delta \,;[L, L']}\,(\text{cut})}{\vdash \Gamma, X^*, X, \Delta \,;[L, L']}\,(\text{cut})}{\vdash \Gamma, \Delta \,;[L, L'],}\,(\text{cut})$$

which has a lesser rank than the original.

Finally we consider the case where the inference (R) is a tensor rule introducing one of the cut formula. In this case the structure of π is as shown below.

$$\dfrac{\dfrac{\dfrac{\vdots}{\vdash \Gamma, X^*, Y^*, X \otimes Y \,;[L]}\,(R')}{\vdash \Gamma, X^* \otimes Y^*, X \otimes Y \,;[L]}\,(\otimes)}{\vdash \Gamma \,;[L]}\,(\text{cut})$$

Unfortunately we cannot proceed as usual by permuting the rule (R') below the cut: for example it may be a mix separating X^* and Y^*. Instead the branch containing $X \otimes Y$ must be rewritten so that the inference introducing the cut-formula occurs last. Let π_0 be the subproof of π whose last inference is (R'). Let (T) be the inference introducing $X \otimes Y$. Clearly (T) occurs in π_0; suppose that is not (R'). Consider the inference immediately below (T); since this inference has a premise, namely the conclusion of (T), it cannot be an occurrence of the axiom or unit rules. Further, by hypothesis, π has only one cut, so π_0 is cut-free.

Hence the rule after (T) must be an occurrence of the exchange, tensor or mix rules. In all cases it will commute with (T). Hence we can rewrite π_0 to an equivalent proof π_0' where (T) is the last inference, giving the following:

$$\dfrac{\dfrac{\dfrac{\dfrac{\vdots}{\vdash \Gamma, X^*, Y^*, X, Y\ ; [L]}}{\vdash \Gamma, X^*, Y^*, X \otimes Y\ ; [L]}\,(\otimes)}{\vdash \Gamma, X^* \otimes Y^*, X \otimes Y\ ; [L]}\,(\otimes)}{\vdash \Gamma\ ; [L].}\,(\text{cut})$$

As in the case above, this can be rewritten to a proof of lesser cut-rank. □

Theorem 4.9 (Soundness of Cut-Elimination)**.** *If π reduces to π' by some number of steps of the cut elimination procedure described above, then $[\![\pi]\!] = [\![\pi']\!]$.*

Proof. Omitted; see soundness of cut-elimination for proof-nets, below. □

Remark. The soundness result refers to the cut-elimination procedure used to prove Theorem 4.8. It's worth noting that there are possible cut elimination procedures, equally good at producing valid proofs, which are not sound. As a simple example, we note that in any cut-free proof of the sequent

$$\vdash \Gamma\ ; [h]$$

the loop h must be introduced by an application of the h-unit rule, and the mix rule. Omitting these inferences yields a perfectly serviceable cut-free proof of $\vdash \Gamma$, however its denotation will not coincide with that of the original.

The sequent calculus presentation is the most natural way to define the semantics of the logic; however, it is far from perfect. As is usual for sequent calculi, the cut elimination procedure is not confluent, so a proof can have many denotationally equivalent normal forms. To remedy these defects, in the next section we introduce proof-nets.

4.3 Proof-nets

In this section we present a graphical proof notation for **mCQL** , closely related to *proof-nets* for multiplicative linear logic [Gir87a]. These proof-nets improve on the sequent calculus in a number of respects: the cut-elimination process is sound and strongly normalising. More importantly, the proof-nets provide a faithful and fully complete representation of $F\mathcal{A}$.

Definition 4.10 (proof-net)**.** A *proof-net* is a finite oriented graph with edges labelled by formulae. The graph is constructed by composing the following nodes, which we call *links*, while respecting the labelling on the incoming and outgoing edges.

Axiom No incoming edges; two out-going edges. The link itself is labelled by an axiom $f : A \to B$. One outgoing edge is labelled A^*, the other, B.

Cut Two incoming edges; no outgoing edges. Each cut is labelled either by an axiom $f : A \to B$ with incoming edges labelled by A and B^*, or else it is labelled by an identity with the incoming edges labelled by X and X^* for an arbitrary formula X.

Times Two incoming edges labelled X and Y; one outgoing edge labelled $X \otimes Y$.

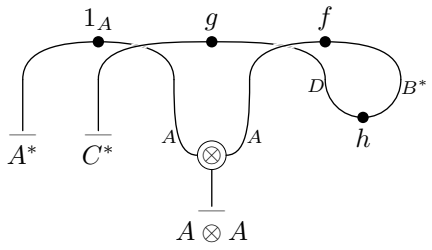
Loop No incoming or outgoing edges. The link is labelled by a *loop* $[f]$, that is, an equivalence class of endomorphisms in $\mathcal{A}$.

The orientation is such that edges enter the node from the top, and exit from the bottom. The *conclusions* of the net are those labels on outgoing edges of links which are left unconnected. The order of the conclusions is significant.

We emphasise that empty net is a valid net, having no conclusions.

Remark. Unlike proof-nets for linear logic, our proof-nets have no correctness criteria [DR89, HvG03]. As we will prove shortly, every **mCQL** proof-net can be translated back into an equivalent sequent proof.

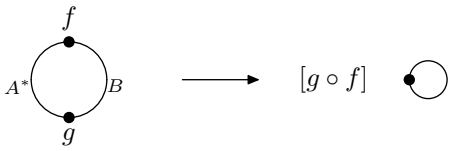
Example 4.11. This net is the translation of Example 4.4.



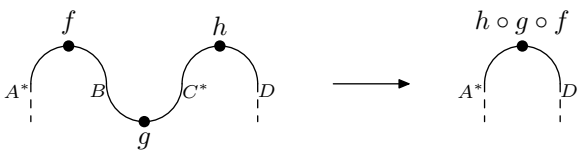
Definition 4.12 (β-Reduction)**.** We define a one-step reduction relation between proof-nets by the following local rewrites on cut links:

1. A cut between atomic formulae. Atomic formulae are only introduced by axiom links, so there are two subcases.

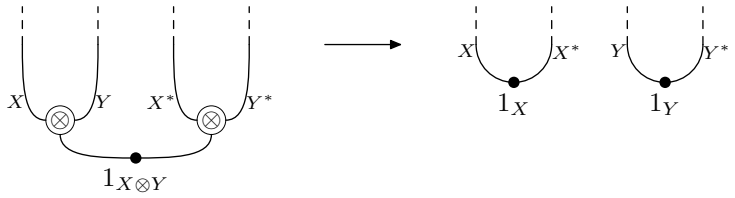
   (a) If both formulae belong to the same axiom (say f):



   (b) If the cut formulae are conclusions of different axioms, say f and h:



2. Cut between two tensor products:



Let $\xrightarrow[\beta]{}$ be the reflexive transitive closure of this one step relation; that is $\pi \xrightarrow[\beta]{} \pi'$ if the proof-net π can be rewritten to π' by a sequence of zero or more of steps shown above. In this circumstance we say that π *beta-reduces* to π'.

Theorem 4.13 (Cut Elimination)**.** *β-reduction for **mCQL** proof-nets is strongly normalising; further, a proof-net is β-normal if and only if it contains no cut links.*

Proof. First, observe that all rewrites reduce the number of links in the proof-net, so there can be no infinite rewrite sequence. Additionally, the left hand sides of the rewrite rules exhaust all the possibilities for introducing a cut, hence if π contains a cut, one of the three rewrites must apply, and so π is not normal. Since the rewrites apply only to cut-links, the β-normal proof-nets are exactly the cut-free proof-nets.

Finally we must show the confluence of β-reduction. The rewrite rules are all purely local, so rewrites cannot interfere with each other unless they overlap. There are two possible overlappings of rewrites: these are shown in Figure 4.6. In the first case associativity in the underlying category $\mathcal{A}$ prevents any conflict; in the second, there is no conflict because the two endomorphisms are cyclic permutations of each other, hence they are both labels for the same loop.

Since the process is terminating and confluent, β-reduction is strongly normalising. □

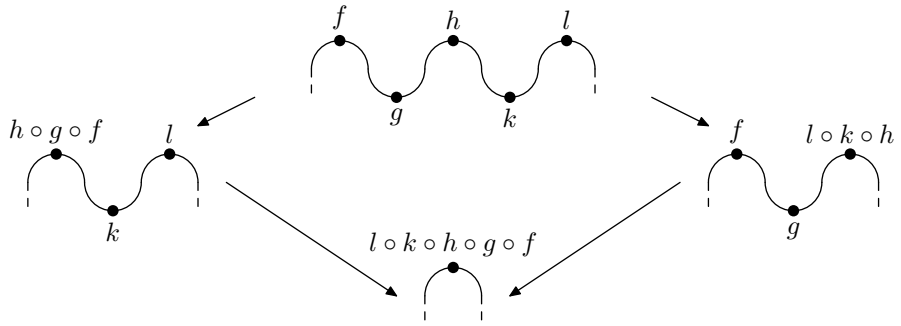
Definition 4.14 (Semantics of proof-nets)**.** Let ν be a proof-net with conclusions Γ. Define an arrow of $F\mathcal{A}$, $[\![\nu]\!] : I \to \bigotimes \Gamma$, by recursion on the structure of ν.

- If ν is just an axiom link corresponding to the arrow $f : A \to B$, then let $[\![\nu]\!] = \ulcorner f \urcorner : I \to A^* \otimes B$.
- If ν has several disconnected components $\nu_1, \ldots, \nu_n$ then define

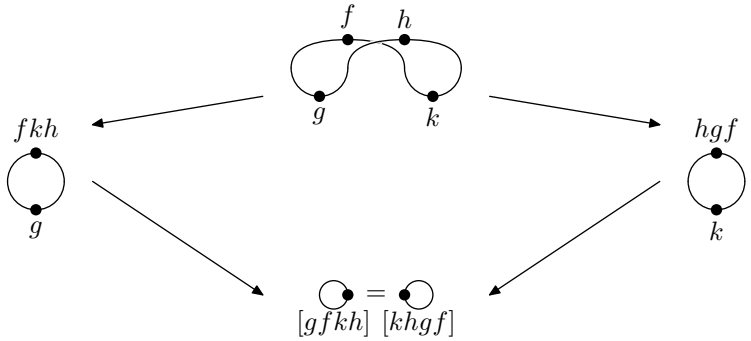
$$[\![\nu]\!] = \bigotimes_{i=1}^{n} [\![\nu_i]\!].$$

- If ν is built by applying a cut labelled by $f : A \to B$ between conclusions A and B^* of ν', suppose that we have constructed $[\![\nu']\!] : I \to \Gamma \otimes A \otimes B^* \otimes \Delta$. Then define $[\![\nu]\!]$ by the composition

$$I \xrightarrow{[\![\nu']\!]} \Gamma \otimes A \otimes B^* \otimes \Delta \xrightarrow{1_\Gamma \otimes \llcorner g \lrcorner \otimes 1_\Delta} \Gamma \otimes \Delta.$$



Resolving divergence between 2 rewrites of type 1(b)



Resolving divergence between rewrites of type 1(a) and 1(b)

Figure 4.6: Confluence of β-reduction

- If ν is built by applying a $\otimes$-link between conclusions A and B of ν' then let $[\![\nu]\!] = [\![\nu']\!]$.
- If ν is a loop labelled by $[f]$, where $f : A \to A$ for some atom A, then $[\![\nu]\!] = \epsilon_{A^*} \circ \ulcorner f \urcorner$. Note that, by Theorem 2.38, this value is independent of the choice of f.
- If ν is the empty net $[\![\nu]\!] = 1_I$.

All these constructions commute wherever the required compositions are defined due to the functoriality of the tensor, hence $[\![\nu]\!]$ is well defined.

Theorem 4.15 (Soundness)**.** *If a net ν reduces to ν' by one or more steps of the cut-elimination procedure of Theorem 4.13 then $[\![\nu]\!] = [\![\nu']\!]$.*

Proof. Each of the rewrite rules of the cut elimination procedure preserves denotation. For each rewrite rule we show the corresponding equation.

1. Suppose we have arrows $B \xrightarrow{e} A \xrightarrow{f} B \xrightarrow{g} C \xrightarrow{h} D$ in $\mathcal{A}$; then
   (a) We have
   $$\begin{array}{rcl} \llcorner e \lrcorner \circ \ulcorner f \urcorner & = & \epsilon_{A^*} \circ (1_{A^*} \otimes e) \circ (1_{A^*} \otimes f) \circ \eta_A \\ & = & \epsilon_{A^*} \circ (1_{A^*} \otimes (e \circ f)) \circ \eta_A \\ & = & \epsilon_{A^*} \circ \ulcorner e \circ f \urcorner \end{array}$$
   directly from the definition of the name and coname.
   (b) The required equation
   $$(1_{A^*} \otimes \llcorner g \lrcorner \otimes 1_D) \circ (\ulcorner f \urcorner \otimes \ulcorner h \urcorner) = \ulcorner h \circ g \circ f \urcorner$$
   is Lemma 2.20 verbatim.
2. The case for tensor follows from $\epsilon_{A \otimes B} = \sigma \circ (\epsilon_A \otimes \epsilon_B)$.

The result follows by the functoriality of the tensor. □

Definition 4.16 (Translation into proof-nets)**.** Given an **mCQL** sequent proof π, we define a proof-net $N\pi$ by recursion over the structure of π.

- If proof π is just an axiom, let $N\pi$ be the proof-net containing just the corresponding axiom link.
- If proof π is a just an application of the h-unit rule for some $h : A \to A$, let $N\pi$ be the proof-net containing a loop labelled by $[h]$.
- If π arises from π' by an application of the g-cut rule for arrow g, form $N\pi$ by adding a cut link labelled by g between the conclusions of $N\pi'$ corresponding to the active formulae of the cut rule.
- Similarly if π is formed from π' by a cut between two compound formulae X and X^*, form $N\pi$ by adding an identity cut link between the conclusions X and X^* of $N\pi'$.
- Suppose π arises from subproofs π_1 and π_2 by the mix rule. Let $N\pi$ be the proof-net formed by juxtaposing $N\pi_1$ and $N\pi_2$.

- If π arises from π' by an application of the times rule, form $N\pi$ by adding a $\otimes$-link between the conclusions of $N\pi'$ corresponding to the active formulae of the tensor rule.

Remark. Proof-nets can be composed by simulating the sequent rules for binary cut via this translation.

Lemma 4.17 (Simulating Cut-Elimination)**.** *Let π, π' be **mCQL** sequent proofs such that π reduces to π' by one step of the cut elimination procedure defined in Theorem 4.8; then $N\pi \xrightarrow[\beta]{} N\pi'$.*

Proof. Observe that rewrites which do not reduce the cut rank of π have no effect at all on $N\pi$ since these are simply reorderings of inferences which do not interact. In that case $N\pi = N\pi'$.

Otherwise, there are three cases where $r(\pi) > r(\pi')$, corresponding to Equations (4.1), (4.2) and (4.3). Each of these rewrites corresponds to β-rules 1(a), 1(b), and 2 respectively, hence $N\pi \xrightarrow[\beta]{} N\pi'$ in one step. □

Proposition 4.18. *Let π be a proof in **mCQL**. Then $[\![\pi]\!] = [\![N\pi]\!]$.*

Proof. Each step of the translation from sequents to proof-nets preserves denotation. □

Since cut-elimination for proof-nets is sound, the preceding two results combine to prove Theorem 4.9, soundness for sequent cut-elimination. Some technical lemmas which characterise the structure of proof-nets now follow.

Lemma 4.19 (Separation)**.** *If π is a cut-free proof of $\vdash \Gamma\ ;[]$ which contains the literals $A_1^*, B_1, \ldots, A_n^*, B_n$ then $[\![\pi]\!]$ has the form*

$$I \xrightarrow{\ulcorner f_1 \urcorner \otimes \cdots \otimes \ulcorner f_n \urcorner} (A_1^* \otimes B_1) \otimes \cdots \otimes (A_n^* \otimes B_n) \xrightarrow{\quad s \quad} \Gamma$$

where the $f_i : A_i \to B_i$ are axioms, and s is a permutation of the components of the tensor.

Proof. Induction on the structure π.

- If π is an axiom then $[\![\pi]\!] = \ulcorner f_1 \urcorner$, giving the result.
- If π is derived from π_1, π_2 by the mix rule, then $[\![\pi]\!] = [\![\pi_1]\!] \otimes [\![\pi_2]\!]$ by construction. By induction hypothesis $[\![\pi_1]\!] = (\bigotimes_{i=1}^{k} \ulcorner f_i \urcorner); s_1$ and $[\![\pi_2]\!] = (\bigotimes_{i=k+1}^{n} \ulcorner f_i \urcorner); s_2$ for some k, which gives the result, since $s_1 \otimes s_2$ is a permutation on the combined tensor.
- If π is derived from π' by the tensor rule then $[\![\pi]\!] = [\![\pi']\!]$, and result follows by induction.
- If π arises by an exchange, then $[\![\pi]\!] = [\![\pi']\!]; \sigma$ where σ is a permutation. Result follows trivially.

□

Note that s is unique up to the initial choice of the ordering of the axioms $f_1, \ldots, f_n$. In the following some canonical ordering will be taken for granted; call s the *induced permutation* of $[\![\pi]\!]$.

A long standing observation[Gir87b, AJ94] in the theory of multiplicative linear logic is that any cut-free **MLL** proof-net π, with only atomic formulae as axioms, can be specified entirely by $\pi \cong (\Gamma, \sigma)$ where Γ is the sequent proven by π and σ is a permutation which is fixpoint-free and a product of disjoint transpositions. Such a permutation may be regarded as a category comprised of a finite coproduct of copies of the category **2**; as described Section 2.8 we refer to such categories as involutions.

Since the proof-net is cut free, its conclusions provide sufficient information to construct the bottom part of the proof net, while the transpositions of σ correspond to the axiom links. In the absence of loops, cut-free **mCQL** proof-nets have the same property, with the distinction that we also require a functor $\theta : \sigma \to \mathcal{A}$ to label the axioms. We can formalise this in the following lemma.

Lemma 4.20. *In order to uniquely specify a normal proof-net, three data are required:*

1. *The list of conclusions* Γ*;*
2. *An involution* σ *on the literals occurring in* Γ*, together with a functor* $\theta : \sigma \to \mathcal{A}$*.*
3. *A multiset* L *of loops in* $\mathcal{A}$*.*

Proof. We divide the proof-net into three parts: the axioms, the logic, and the loops. Each item on the list uniquely specifies one part of the proof-net.

Each conclusion A of Γ determines the structure of the net up to the axiom links which introduced its atoms. If $A = B \otimes C$ then its parent link is a $\otimes$-link, which in turn has a left and right premise. If A is a literal then its parent is an axiom, so we are done. Since Γ is itself ordered this procedure puts an ordering on the literals occurring among the conclusions.

Given this ordering, the involution σ connects pairs of literals to show the position of the axiom links. The functor θ maps each such axiom link to an axiom of $\mathcal{A}$ giving their labels.

All the parts of the net which deduce the conclusions are now defined. Each loop of L defines a normal loop in the obvious fashion to complete the construction. □

Consider an **mCQL** proof π which is constructed solely by the mix of axioms

$$\frac{}{\vdash A_i^*, B_i.}\ (f_i\text{-axiom})$$

In this case $[\![\pi]\!] = \ulcorner f_1 \urcorner \otimes \cdots \otimes \ulcorner f_n \urcorner$ and the induced permutation is the identity. The resulting proof-net $N\pi$ is also just the mix of axiom links: the permutation corresponding to those axioms is

$$(1\ \ 2) \cdots (2n-1\ \ 2n)\ .$$

Denote this permutation ϱ_n.

Lemma 4.21 (Characterisation)**.** *Let π be a cut free **mCQL** proof of $\vdash \Gamma\,;[]$ containing n axioms, such that $N\pi \cong (\Gamma;\sigma,\theta,\emptyset)$ as described above. If s is the permutation induced by $[\![\pi]\!]$ then $\sigma = s^{-1}\varrho_n s$.*

Proof. We use induction on the structure of π. We simplify our notation by omitting the loops from the **mCQL** sequents, since they are empty in all cases.

- π is an axiom : then $N\pi \cong (\vdash A^*, B\,;\varrho_1)$. In this case $[\![\pi]\!] = \eta_A$ and the induced permutation s is just the identity, so trivially $\sigma = s^{-1}\varrho_1 s$.

- π is formed by π' by application of the tensor rule: We have

$$N\pi' \cong (\vdash \Gamma, A, B, \Delta\,;\sigma')$$
$$N\pi \cong (\vdash \Gamma, A \otimes B, \Delta\,;\sigma'),$$

  and $[\![\pi]\!] = [\![\pi']\!]$ by construction. By induction hypothesis, the result holds for π' and hence also for π.

- π is formed by applying the mix rule to π_1 and π_2: in this case

$$N\pi_1 \cong (\vdash \Gamma\,;\sigma_1)$$
$$N\pi_2 \cong (\vdash \Delta\,;\sigma_2)$$
$$N\pi \cong (\vdash \Gamma, \Delta\,;\sigma_1\sigma_2),$$

  modulo a suitable renaming of the atoms of π_2. By construction $[\![\pi]\!] = [\![\pi_1]\!] \otimes [\![\pi_2]\!]$. Since the induced permutations s_1 and s_2 are disjoint, the permutation induced by $[\![\pi_1]\!] \otimes [\![\pi_2]\!]$ is just their product $s = s_1 s_2$. Now by induction hypothesis $\sigma_1 = s_1^{-1}\varrho_k s_1$ and $\sigma_2 = s_2^{-1}\varrho_{k'} s_2$, hence

$$\sigma_1\sigma_2 = (s_1^{-1}\varrho_k s_1)(s_2^{-1}\varrho_{k'} s_2) = s_2^{-1}s_1^{-1}\varrho_k\varrho_{k'} s_1 s_2$$

  which gives the result.

- π arises from π' by the exchange rule. It suffices to consider the case of a transposition of two atoms, A, B, since more general permutations can be constructed from products of transpositions. Let

$$N\pi' \cong (\vdash \Gamma, A, B, \Delta\,;\sigma')$$

  so that π is a proof of $\vdash \Gamma, B, A, \Delta$. If i is the index of A in π, then

$$\sigma' = (a\ i)(b\ i+1)\sigma''$$

  and

$$\sigma\ = (a\ i+1)(b\ i)\sigma''$$

  for some a, b, σ''. By its construction $[\![\pi]\!] = [\![\pi']\!]; t$ where $t = (i\ i+1)$, and hence the permutation induced by $[\![\pi]\!]$ is $s = s't$, where s' is the induced permutation of π'. But note that

$$\sigma = (i\ i+1)\sigma'(i\ i+1) = ts'^{-1}\varrho_n s' t = s^{-1}\varrho s$$

  which is the desired result.

□

Proposition 4.22 (Sequentialisation)**.** *For all normal proof-nets ν there exists a sequent proof $P\nu$ such that $\nu = NP\nu$.*

Proof. We will define $P\nu$ by recursion over the structure of ν, and establish the $NP\nu = \nu$ in each case by induction.

If ν consists of disconnected components ν_1,ν_2, $\ldots$, ν_n with conclusions $\Gamma_1, \Gamma_2, \ldots, \Gamma_n$ respectively then $P\nu$ is constructed by combining $P\nu_1, P\nu_2, \ldots, P\nu_n$ via the mix rule, as shown:

$$\dfrac{\dfrac{\dfrac{\vdots\, P\nu_1}{\vdash \Gamma_1} \qquad \dfrac{\vdots\, P\nu_2}{\vdash \Gamma_2}}{\vdash \Gamma_1, \Gamma_2}\,(\text{mix}) \qquad \vdots \qquad \dfrac{\vdots\, P\nu_n}{\vdash \Gamma_2}}{\vdash \Gamma_1, \Gamma_2, \ldots, \Gamma_n}\,(\text{mix})$$

By induction for each subproof $P\nu_i$ we have $NP\nu_i = \nu_i$; it is immediate from the definition of $NP\nu$ that the equality holds for the construction above.

If ν consists of a loop labelled by $[h]$ then let $P\nu$ be the proof $\dfrac{}{\vdash ; [h].}\,(h\text{-unit})$ Again we have $NP\nu = \nu$ immediately from the definition.

Otherwise, assume that ν is presented as $\nu \cong (\Gamma, \sigma, \theta)$ so that the lower part of the proof-net is planar. Suppose further that the axioms occurring in ν are $f_i : A_i \to B_i$ for $i = 1, \ldots, n$, ordered by the order that their left conclusions occur as subformulae of Γ, so that

$$\sigma = \prod_{i=1}^{n} (a_i\ b_i) \quad \text{where} \quad \begin{array}{l} a_i < b_i \\ i < j \Leftrightarrow a_i < a_j. \end{array}$$

Then form the proof

$$\dfrac{\dfrac{\dfrac{}{\vdash A_1^*, B_1}\,(f_1\text{-axiom}) \qquad \dfrac{}{\vdash A_2^*, B_2}\,(f_2\text{-axiom})}{\vdash A_1^*, B_1, A_2^*, B_2}\,(\text{mix}) \qquad \vdots \qquad \dfrac{}{\vdash A_n^*, B_n}\,(f_n\text{-axiom})}{\vdash A_1^*, B_1, A_2^*, B_2, \ldots, A_n^*, B_n}\,(\text{mix})$$

by mixing together the f_i-axioms for all i. Call this proof π_1. Let Γ' be the sequent obtained from Γ by erasing every occurrence of the symbol $\otimes$ and replacing it with a comma. We note that the difference between Γ' and the conclusion of π_1 is simply a permutation of literals. Let $P\nu$ be the proof

$$\dfrac{\dfrac{\vdots\, \pi_1}{\vdash A_1^*, B_1, A_2^*, B_2, A_n^*, B_n}}{\vdash \Gamma'}\,(\text{exch})$$
$$\vdots\, \pi_2$$
$$\vdash \Gamma$$

where π_2 is a subproof consisting entirely of repeated applications of the tensor rule. The exchange rule is via the permutation

$$s = \begin{pmatrix} \cdots & 2i-1 & 2i & \cdots \\ \cdots & a_i & b_i & \cdots \end{pmatrix}.$$

Note that s is exactly the induced permutation of $[\![P\nu]\!]$, hence $NP\nu \cong (\Gamma, s^{-1}\varrho_n s, \theta')$. Now a simple calculation shows that, for $i = 1 \ldots n$,

$$s^{-1}(2i-1\ 2i)s = (a_i\ b_i),$$

hence $\sigma = s^{-1}\varrho_n s$. Since θ' sends the ith component of σ to f_i, $\theta' = \theta$, hence $NP\nu = \nu$. □

Corollary 4.23. $[\![P\nu]\!] = [\![\nu]\!]$.

Proof. Since $\nu = NP\nu$, we have $[\![\nu]\!] = [\![NP\nu]\!]$, and by Proposition 4.18 $[\![NP\nu]\!] = [\![P\nu]\!]$. □

To complete the treatment of **mCQL**, we now show that **mCQL** proof-nets are a faithful and fully complete model of the category $F\mathcal{A}$.

Theorem 4.24 (Faithfulness)**.** *Two nets ν, ν' with the same conclusions Γ have the same normal form if and only if $[\![\nu]\!] = [\![\nu']\!]$.*

Proof. Since cut-elimination is sound, we need only consider the case where ν, ν' are already normal. Up to a scalar factor, $[\![\nu]\!]$ must have the following structure,

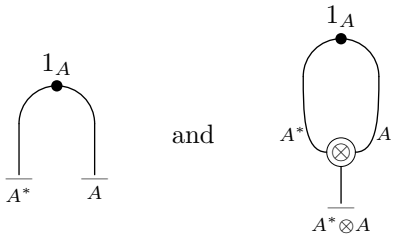
$$I \xrightarrow{\ulcorner f_1 \urcorner \otimes \cdots \otimes \ulcorner f_n \urcorner} \bigotimes_{i=1}^{n} (A^*_{2i-1} \otimes A_{2i}) \xrightarrow{\ s\ } \Gamma$$

where s is the induced permutation. Given Γ, and fixing the order the names $\ulcorner f_i \urcorner$, s suffices to determine the loop-free part of the net.

Let z be a scalar of $F\mathcal{A}$. Then $z = \bigotimes_i [\![h_i]\!]$ for some possibly empty family of loops $\{h_i\}_i$. Note that $z \otimes f = f \otimes z$ for all arrows f in $F\mathcal{A}$, so any scalar determines a multiset of loops. Further, since $F\mathcal{A}$ is freely constructed, we have only structural equalities between scalars, hence this representation is unique.

Therefore, by Lemma 4.20, $[\![\nu]\!]$ provides all the data required to characterise ν exactly, and so if $[\![\nu]\!] = [\![\nu']\!]$ then $\nu = \nu'$. □

It should be noted that the faithfulness result required the conclusions of the nets to be specified. In fact the syntax is not truly injective onto the arrows of $F\mathcal{A}$. For example, the nets



both denote the map η_A.

Recall that Theorem (2.8) allows the arrows to characterises the arrows of $F\mathcal{A}$ by an involution σ, a functor $\theta : \sigma \to \mathcal{A}$ and a multiset of loops, L.

Lemma 4.25. *In a $F\mathcal{A}$, let $f : A \to B \equiv (\sigma, \theta, L)$ such that the arrows in the image of θ are $\{f_i\}_i$; then $\ulcorner f \urcorner = \lambda \bullet (s \circ \bigotimes_i \ulcorner f_i \urcorner)$ for some permutation s and scalar factor λ.*

Proof. An arrow and its name have the same representation in terms of labelled involutions and loops, hence f and $\ulcorner f \urcorner$ have the same scalar factors. By proposition 8.1 of [KL80] we have that any arrow[2] $f : X \to Y$ in $F\mathcal{A}$ is of the form

$$X \xrightarrow[x]{} X_1 \otimes \cdots \otimes X_n \xrightarrow[g_1 \otimes \cdots \otimes g_n]{} Y_1 \otimes \cdots \otimes Y_n \xrightarrow[y]{} Y,$$

where x, y are structural isomorphisms and each g_i is either $g, g^*, \ulcorner g \urcorner$ or $\llcorner g \lrcorner$ for arrows g of $\mathcal{A}$, or otherwise is a scalar. In particular, when f is itself a name then $X = I$ and all the g_i must be names or scalars. Since $\ulcorner \lambda \bullet f \urcorner = \lambda \bullet \ulcorner f \urcorner$ this suffices. □

Theorem 4.26 (Full Completeness)**.** *Let $f : X \to Y$ be an arrow of $F\mathcal{A}$, the free compact closed category on $\mathcal{A}$; then there exists a proof-net ν such that $\ulcorner f \urcorner = [\![\nu]\!]$.*

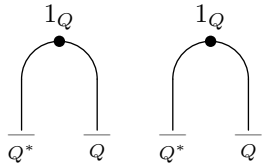
Proof. By Kelly-Laplaza $\ulcorner f \urcorner \approx (\sigma, \theta, L)$. By the preceding lemma $\ulcorner f \urcorner = \lambda \bullet (s \circ \bigotimes_i \ulcorner f_i \urcorner)$; Let ν be the normal proof-net given by $(\vdash X^* \otimes Y, s^{-1}\varsigma_n s, \theta, L)$. By Lemma 4.21 $[\![\nu]\!] = \ulcorner f \urcorner$. □

4.4 Example: Entanglement Swapping

To illustrate how **mCQL** proof-nets can be used in the context of quantum computing, we now consider a simple protocol called *entanglement swapping* [ZZHE93], here shown in a post-selected variant, which is a variant of the teleportation protocol described in Chapter 1.

In the teleportation protocol we had to assume that Alice and Bob shared an entangled pair before they could carry out the protocol. Entanglement swapping allows the parties to establish such a shared state via an intermediary who acts as a "quantum telephone exchange". We will describe this protocol using proof-nets over the category Pauli; this has only one object $\mathbb{C}^2 = Q$, its arrows are the Pauli group.

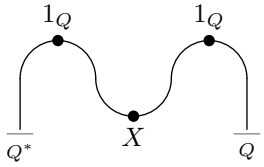
Initially, both Alice and Bob share a Bell pair with the Exchange; to keep things simple we'll take them both as the state $|\beta_0\rangle = |00\rangle + |11\rangle$. We represent the initial situation as a pair of axiom links labelled by 1_Q.



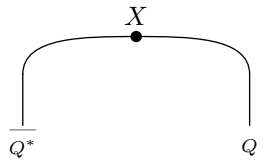
(Since the proof-net notation does not represent the participants in the protocol we make the convention that the left most formula belongs to Alice, the right most to Bob, and the others to the Exchange.) In order to connect Alice and Bob, the Exchange then performs a Bell basis measurement on the qubits in its

[2] In our setting we consider only those objects which Kelly and Laplaza call *reduced*.

possession. Supposing the outcome was $|01\rangle + |10\rangle = \ulcorner X \urcorner$, we represent this as an X-cut between the qubits:



We obtain the final state of the protocol by performing cut-elimination:



Hence Alice and Bob share an entangled pair, which they can now use for teleportation, provided they correct the X error.

While this example is very simple, indeed the simplest possible, it demonstrates the basic ingredients of the proof-net approach. First, the different interactions which make up the protocol are put together in the form of a net, with axioms for state preparations and cuts for measurements. The proof-net is then normalised to give the final state of the system.

It is worth emphasising that this proof-net represents only one possible run of the protocol; a complete description would require all the possible measurement outcomes to be accounted for, and for classical communication between the participants in order to deal with the errors. These issues are not discussed in this thesis; one possible approach is presented in [AD06].

Chapter 5

MLL and Entanglement

In this chapter we examine some formal characterisations of entangled states in abstract quantum mechanics carried out in strongly compact closed categories. We will consider only the bipartite case here: multipartite entanglement is considerably more complicated, and a discussion of it is postponed until the following chapter.

Entanglement is a fundamental phenomenon in quantum computation. Loosely speaking, a pair of quantum systems is entangled if the state of the joint system cannot be expressed as a pair of states of its subsystems. Exploitation of such non-local correlations lies at the heart of many quantum communication protocols — see [BBC$^+$93, Eke91b, FGM01, DP05] for some examples — and it is implicated in the speedup observed in quantum algorithms [JL03].

Gisin [Gis91] proved that any bipartite pure state which is not a product will violate some version of the Bell inequalities. This sharp division between the entangled and unentangled, or separable, states is strengthened in [BBPS96] which shows that an ensemble of weakly entangled states can be distilled into a smaller number of maximally entangled states. In this sense, any bipartite entangled state is equivalent to any other; the separable states form an intrinsically different class.

Given the above, it would be desirable in a quantum programming language to distinguish between separable and entangled states at the level of types. Unfortunately the set of entangled states in a Hilbert space is not closed under addition, and therefore does not form a subspace. Hence the objects of **FDHilb**, and by analogy any compact closed category, cannot provide the desired type information.

Describing the use of entangled resources in [BBPS96] Bennett et al remark:

> Note that qubits are a directed channel resource, sent in a particular direction from the sender to the receiver; by contrast, ebits [entangled pairs] are an undirected resource shared between sender and receiver.

The notion of a bidirectional channel also arises in linear logic [Gir87a], where the multiplicative disjunction $A \parr B$ can be understood as the simultaneous implications $\neg A \Rightarrow B$ and $A \Leftarrow \neg B$. We will show that multiplicative linear types do provide the extra information needed to accurately describe separable and entangled states. Since any compact closed category is also *-autonomous

[Bar79], we already have a model of multiplicative linear logic[1] (**MLL**) [Bar91, See89]. However, compact closed models are degenerate: the connectives $\otimes$ and $\parr$ are identified, so more work is required.

In the next section, we abstract from the standard descriptions of entangled states in the Hilbert space formalism to give characterisations of separable, entangled, and maximally entangled in an arbitrary compact closed category. In Section 5.2 we introduce *double gluing* [Loa94, Tan97, HS03], a construction which produces a non-degenerate *-autonomous category from any compact closed category. Double-gluing is partly successful in separating the entangled and separable states. Finally, we consider the syntactic relations between **mCQL** and **MLL** and prove that if an **mCQL** proof is typable in **MLL** then its type determines whether or not it is entangled.

The notions of separability and entanglement are both relative to a given partition of the state space into some number of subspaces. For example, a pair of Bell states may be considered as separable or maximally entangled, purely based on which sets of qubits are counted together as one subsystem. For this reason we will temporarily treat the tensor structure as non-strict: the associativity of the tensor will describe the division of a given object into two sub-objects.

5.1 Entangled States

Let $|\psi\rangle \in A \otimes B$ be a quantum state. To any such pure state there exists a *Schmidt decomposition*: that is, sequences of orthonormal vectors $a_i \in A$ and $b_i \in B$ such that

$$|\psi\rangle = \sum_i \lambda_i \, |a_i\rangle \, |b_i\rangle$$

where the *Schmidt coefficients* λ_i are non-negative reals such that $\sum_i \lambda_i^2 = 1$. (See, e.g. [NC00] for details). The number of non-zero Schmidt coefficients in the decomposition provides a crude measure of the entanglement of a bipartite state. Given the Schmidt decomposition above, the reduced density matrices of the two subsystems of $|\psi\rangle$ are respectively

$$\rho_A = \sum_i \lambda_i^2 \, |a_i\rangle \, \langle a_i| \quad \text{and} \quad \rho_B = \sum_i \lambda_i^2 \, |b_i\rangle \, \langle b_i| \, . \tag{5.1}$$

Another consequence is that if $\dim_B > \dim_A$ then the correlations between the two subsystems are limited to a no greater than $\dim_A$ dimensional subspace of B [Eke91a]. From a more abstract point of view, this fact is contained in the isomorphism between $A \otimes B$ and the space of linear maps from A to B; clearly any such map f has $\dim(\mathrm{Im}(f)) \leq \dim_A$. We now examine the relationship between the map coded by a bipartite state and the entanglement between its subsystems.

Definition 5.1. If there exist quantum states $|\psi_A\rangle \in A$ and $|\psi_B\rangle \in B$ such that

$$|\psi\rangle = |\psi_A\rangle \otimes |\psi_B\rangle$$

then $|\psi\rangle$ is called *separable*; otherwise $|\psi\rangle$ is *entangled*.

[1] We speak here only of multiplicative linear logic without units.

Lemma 5.2. *Let $f : A \to B$ be a linear map such that $\ulcorner f \urcorner(1) = \psi_A \otimes \psi_B$, i.e. a separable state. Then*

$$f = |\psi_B\rangle \langle\psi_A| .$$

Proof. Let $\{|a_i\rangle\}_i$ be a basis for A^* such that $|\psi_A\rangle = \sum_i \lambda_i |a_i\rangle$. Then

$$\ulcorner f \urcorner(1) = (\sum_i \lambda_i |a_i\rangle) \otimes |\psi_B\rangle$$

from whence

$$f : |\overline{a_i}\rangle \mapsto \lambda_i \sum_j \lambda_j |b_j\rangle .$$

The map $|\overline{a_i}\rangle \mapsto \lambda_i$ is the equivalent to the definition of $\langle\psi_A|$, so we are done. □

The class of maps $|\psi_B\rangle \langle\psi_A|$ exhibit the weakest possible dependency between input and output for a non-trivial linear map. The only influence of the argument on the result is a magnitude; if we consider the source and target to be quantum states then renormalisation removes even this. Viewing bipartite states as maps, separable states are therefore functions which preserve no information. At the other end of the spectrum we consider maximally entangled states.

For mixed states, entanglement is a complex and subtle property. Many non-equivalent *measures of entanglement* have been proposed to address different regimes, see for example [Eke91a, BBPS96, VPRK97]. For the simple case considered here — bipartite entanglement of pure states — it has been proven that almost[2] all such measures of entanglement are equivalent to the von Neumann entropy of the reduced density operator. Further, a density matrix ρ maximises this function exactly when it is a scalar multiple of the identity. We will adopt this as the concrete definition of maximal entanglement.

Definition 5.3. A pure bipartite quantum state $|\psi\rangle \in A \otimes B$ is *maximally entangled* if its reduced density matrices ρ_A, ρ_B are $\mathbb{1}/n$ and $\mathbb{1}/m$ respectively, where $\dim_A = n$ and $\dim_B = m$.

Note that, due to Eq.(5.1), maximal entanglement by this definition is only possible when $\dim_A = \dim_B$.

The following result is originally due to Hines [Hin04], though the proof here is new.

Lemma 5.4. *Let $dim_A = dim_B = n$; then $f : A \to B$ is unitary iff $\ulcorner f \urcorner(\frac{1}{\sqrt{n}})$ is maximally entangled.*

Proof. Let a_i, b_i be orthonormal bases for A, B. Then,

$$\begin{aligned}
\ulcorner f \urcorner(\frac{1}{\sqrt{n}}) &= (1_{A^*} \otimes f) \circ \eta_A(\frac{1}{\sqrt{n}}) \\
&= \frac{1}{\sqrt{n}}(1_{A^*} \otimes f)(\sum_i a_i^* \otimes a_i) \\
&= \frac{1}{\sqrt{n}} \sum_i a_i^* \otimes f a_i.
\end{aligned}$$

[2] See [VP98] for discussion and an exception.

Since f is unitary, fa_i forms an orthonormal basis, and via this Schmidt decomposition, we have

$$\rho_{A^*} = \frac{1}{n}\sum_i |\, a_i^*\rangle\, \langle a_i^*| = \frac{1}{n}\mathbb{1}$$

and

$$\begin{aligned}\rho_B &= \frac{1}{n}\sum_i |\, fa_i^*\rangle\, \langle fa_i^*| \\ &= \frac{1}{n}\sum_i f\,|\, a_i^*\rangle\, \langle a_i^*|\, f^\dagger \\ &= \frac{1}{n} f\mathbb{1} f^\dagger \\ &= \frac{1}{n}\mathbb{1},\end{aligned}$$

hence the state is maximally entangled.

Conversely, if $|\psi\rangle$ is maximally entangled, suppose its Schmidt decomposition is

$$|\psi\rangle = \sum_i \lambda_i\, |a_i^*\rangle \otimes |b_i\rangle\,,$$

then we have $\rho_{A^*} = \frac{1}{n}\sum_i |\, a_i^*\rangle\, \langle a_i^*| = \frac{1}{n}\mathbb{1}$ which implies that $\lambda_i = \frac{1}{\sqrt{n}}$. Then $f : |a_i\rangle \mapsto |b_i\rangle$ defines a unitary map such that $\ulcorner f \urcorner(\frac{1}{\sqrt{n}}) = |\psi\rangle$. □

Remark. Evaluating $\ulcorner f \urcorner$ at $\frac{1}{\sqrt{n}}$ gives the correct normalisation, but is slightly redundant, since there is only one quantum state in the ray defined by $\ulcorner f \urcorner$.

The preceding lemmas rephrase the characteristics of entangled states as properties of maps; properties which make sense in any strongly compact closed category. Hence we have an abstract definition of the three classes of bipartite states.

Definition 5.5. Let $\mathcal{C}$ be strongly compact closed, and let $\psi = \ulcorner f \urcorner : I \to A^* \otimes B$ be a point. Call ψ *separable*, or *unentangled*, if there exist $\psi_{A^*} : I \to A^*$ and $\psi_B : I \to B$ such that $f = \psi_{A^*}^\dagger \circ \psi_B$; otherwise ψ is *entangled*. Call ψ *maximally entangled* if f is unitary.

Recall (Lemma 2.36) that, in any strongly compact closed category, if a unitary map $f : A \to B$ exists between any objects A, B then $\dim_A = \dim_B$, which correlates nicely with our earlier remarks.

Example 5.6. In **Rel**, the category of sets and relations, a separable state on $A \otimes B$ is the Cartesian product $\psi_A \times \psi_B$ of two subsets $\psi_A \subseteq A, \psi_B \subseteq B$. Since in **Rel** all isomorphisms are unitary, a maximally entangled state is the graph of a bijection between A and B.

5.2 Double Gluing

Since compact closed structure alone cannot provide types which distinguish the entangled and separable states of a given state space, we turn our attention now to a construction which will provide the desired structure.

Every compact closed category is a model of multiplicative linear logic but a degenerate one — the $\otimes$ and $\parr$ connectives are identified. *Double gluing* was introduced by Loader [Loa94] to construct a fully complete model of multiplicative linear logic; see also [Tan97]. This construction provides a general method for producing a *-autonomous category from a compact closed category such that the formulae $\mathbf{A} \otimes \mathbf{B}$ and $\mathbf{A} \parr \mathbf{B}$ denote distinct objects. The following definition and theorem are from [HS03], specialised to the compact closed setting.

Definition 5.7. Suppose $\mathcal{C}$ is compact closed, then the *Double Gluing on* $\mathcal{C}$, $G(\mathcal{C})$, is the category whose objects are triples $\mathbf{A} = (A, U, X)$ where

- A is an object of $\mathcal{C}$
- $U \subseteq \mathcal{C}(I, A)$ is the set of *points* of $\mathbf{A}$.
- $X \subseteq \mathcal{C}(A, I)$ is the set of *co-points* of $\mathbf{A}$.

and whose arrows $\mathbf{f} : (A, U, X) \to (B, V, Y)$ are arrows of $\mathcal{C}$ $f : A \to B$ such that:

- $\forall u \in U \ f \circ u \in V$,
- $\forall y \in Y \ y \circ f \in X$.

In the following, the isomorphism $\mathcal{C}(A, I) \cong \mathcal{C}(I, A^*)$ will be used freely; in particular to write copoints $A \to I$ as arrows $I \to A^*$.

Theorem 5.8. *Given* $\mathbf{A} = (A, U, X)$ *and* $\mathbf{B} = (B, V, Y)$, $G(\mathcal{C})$ *as above is* *-*autonomous with the following structure:*

- $\mathbf{I} = (I, \{1_I\}, \mathcal{C}(I, I))$
- $\mathbf{A}^\perp = (A^*, X, U)$
- $\mathbf{A} \otimes \mathbf{B} = (A \otimes B, W, Z)$ *where*

$$W = \{I \cong I \otimes I \xrightarrow{u \otimes v} A \otimes B \mid u \in U, v \in V\},$$
$$Z = \left\{ A \otimes B \xrightarrow{\llcorner f \lrcorner} I \;\middle|\; \begin{array}{l} \forall u \in U \;\; I \xrightarrow{u} A \xrightarrow{f} B^* \in Y, \\ \forall v \in V \;\; I \xrightarrow{v} B \xrightarrow{f^*} A^* \in X \end{array} \right\}.$$

With $\mathbf{A}$ and $\mathbf{B}$ as above, we define $\mathbf{A} \parr \mathbf{B} = (\mathbf{A}^\perp \otimes \mathbf{B}^\perp)^\perp = (A \otimes B, Z', W')$ where,

$$Z' = \left\{ I \xrightarrow{\ulcorner g \urcorner} A \otimes B \;\middle|\; \begin{array}{l} \forall x \in X \;\; I \xrightarrow{x} A^* \xrightarrow{g} B \in V, \\ \forall y \in Y \;\; I \xrightarrow{y} B^* \xrightarrow{g^*} A \in U \end{array} \right\},$$
$$W' = \{A \otimes B \xrightarrow{x \otimes y} I \otimes I \cong I \mid x \in X, y \in Y\}.$$

The double-gluing construction produces a non-degenerate *-autonomous structure: in general $\mathbf{A} \otimes \mathbf{B} \neq \mathbf{A} \parr \mathbf{B}$.

Since the set of points of $\mathbf{I}$ is a singleton, for each object $\mathbf{A} = (A, U, X)$ we have

$$G\mathcal{C}(\mathbf{I}, \mathbf{A}) \cong U \subseteq \mathcal{C}(I, A)$$

so the points of $\mathbf{A}$ in $G\mathcal{C}$ can be safely interpreted as points in $\mathcal{C}$. The question is: what can be gleaned from the extra type information provided by the gluing construction?

An immediate observation is that the points of $\mathbf{A} \otimes \mathbf{B}$ contain only of separable states of $A \otimes B$; indeed if $\mathbf{A}$ and $\mathbf{B}$ are maximal then the sets of separable states coincides with the points of $\mathbf{A} \otimes \mathbf{B}$. We would hope, conversely, that the points of $\mathbf{A} \parr \mathbf{B}$ consist of entangled states from $A \otimes B$, but this does not hold in general. Let

$$\mathbf{A} = (A,\, \mathcal{C}(I, A),\, \mathcal{C}(A, I))$$
$$\mathbf{B} = (B,\, \mathcal{C}(I, B),\, \mathcal{C}(B, I))$$

and suppose that we have $a : I \to A$ and $b : I \to B$; then $b \circ a^*$ defines a map $A^* \to B$ such that, for any copoint ψ of $\mathbf{A}$, the composite $b \circ a^* \circ \psi$ is a scalar multiple of b, and hence is in $\mathcal{C}(I, B)$; similarly $(b \circ a^*)^*$ maps $\mathcal{C}(B, I)$ to $\mathcal{C}(I, A)$ and hence $\ulcorner b \circ a^* \urcorner$ is a point of $\mathbf{A} \parr \mathbf{B}$ despite being separable.

Since the points and copoints of $\mathbf{A}$ and $\mathbf{B}$ are the the entire hom-sets, the above argument will go through for any map $I \to A \otimes B$; hence the points of $\mathbf{A} \parr \mathbf{B}$ are *all* bipartite states.

Therefore, in the most general case, the double gluing construction gives a proper inclusion of the points of $\mathbf{A} \otimes \mathbf{B}$ into the points of $\mathbf{A} \parr \mathbf{B}$. We must interpret $\mathbf{A} \otimes \mathbf{B}$ as *surely separable* and $\mathbf{A} \parr \mathbf{B}$ as *possibly entangled*. While not entirely satisfactory, this arrangement will reappear in the extended setting discussed in Chapter 6. In general, it is necessary to over-approximate entanglement. However, note that the counter example above depends on the choice of particular sets of points and copoints. We can do better by restricting to a subcategory determined by those points representable by proof-nets of multiplicative linear logic.

5.3 Multiplicative Linear Logic

5.3.1 Sequent Calculus

We make two modification to the standard presentation of multiplicative linear logic. Firstly, we generalise the axiom rule to range over the arrows of a category $\mathcal{A}$ which is not necessarily discrete; hence the axioms may be non-trivial arrows. Since they form a category, we can always eliminate cut in the standard way, by composition in $\mathcal{A}$, as shown in the preceding chapter.

Secondly, and as a consequence, we interpret the atoms as constants rather than as variables, so that the interpretation of a proof will be an arrow in a particular category rather than a dinatural transformation (cf. [GSS91, Blu93, Tan97]). Indeed, the category will be the free compact closed category on $\mathcal{A}$. This is more a matter of presentation than of substance since the underlying structure can in any case be lifted to the more general setting.

Finally, we restrict our attention to the cut-free fragment of **MLL**.

Definition 5.9 (Formulae of **MLL**)**.** The formulae of multiplicative linear logic are generated by a set of literals $A, A^\perp, B, B^\perp, C, C^\perp, \ldots$ and binary connectives $X \otimes Y$ and $X \parr Y$.

The linear negation $(\cdot)^\perp$ is defined on compound formulae by de Morgan duality:

$$(X \otimes Y)^\perp = X^\perp \parr Y^\perp, \qquad (X \parr Y)^\perp = X^\perp \otimes Y^\perp, \qquad X^{\perp\perp} = X.$$

Definition 5.10 (Sequent calculus for **MLL**)**.** A sequent of **MLL** is of the form $\vdash \Gamma$ where Γ is a list of **MLL**-formulae. The inference rules of **MLL** are shown in Fig. 5.1.

$$\frac{}{\vdash A^\perp, B}\,(f\text{-axiom}) \qquad \frac{\vdash \Gamma, X, Y, \Delta}{\vdash \Gamma, Y, X, \Delta}\,(\text{exch})$$

$$\frac{\vdash \Gamma, X, Y, \Delta}{\vdash \Gamma, X \parr Y, \Delta}\,(\parr) \qquad \frac{\vdash \Gamma, X \qquad \vdash Y\Delta}{\vdash \Gamma, X \otimes Y, \Delta}\,(\otimes)$$

Figure 5.1: Inference rules for **MLL** : the parameter of the f-axiom rule ranges over all the arrows of $\mathcal{A}$. Reclaim the usual linear logic axiom by restricting to $f = 1_A$.

By comparison of Fig. 5.1 with Fig. 4.3, it is clear that all the inference rules of **MLL** are derivable in **mCQL**, if $\otimes$ and $\parr$ are identified. The same is true for proof-nets.

5.3.2 Proof-nets

Definition 5.11. An **MLL** *proof-structure* consists of a connected graph whose edges are labelled with **MLL** formulae, and whose vertices – called links – are labelled with the inference rules (except for exchange, which is implicit) such that the following conditions hold:

f-axiom If a link is labelled by the rule f-axiom, for some arrow $f : A \to B$ of $\mathcal{A}$, then it has two outgoing edges labelled respectively by $A^\perp$ and B. It has no incoming edges.

tensor/par If a link is labelled by $\otimes$ or $\parr$ it has two incoming edges, labelled by some formulae X, Y and one outgoing edge labelled either by $X \otimes Y$ or $X \parr Y$ as appropriate.

A proof-structure is a *proof-net* if it arises from the translation of a sequent proof. Since not every proof-structure represents a sequent proof, proof-nets are identified by a correctness criterion, for example the well known Danos-Regnier condition [DR89].

Definition 5.12. A *switching* in an **MLL** proof-structure is a subgraph determined by removing one incoming edge from every $\parr$-link.

Criterion (Danos-Regnier)**.** A proof-structure is an **MLL** proof-net if every switching is acyclic and connected.

Naively a proof-structure is a proof-net if it is constructed from a collection of axiom links by adjoining new links such that:

- if a $\parr$-link is added, both of its incoming edges must come from the same connected component of the structure;
- if a $\otimes$-link is added then its incoming edges must originate in disjoint connected components.

These conditions mirror the rules of the sequent calculus, and are illustrated in Fig. 5.2.

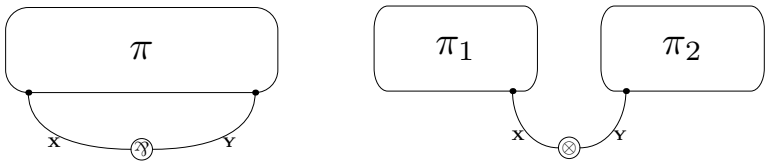
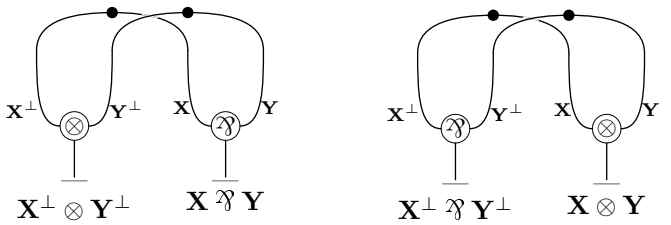


Figure 5.2: Schema for constructing a correct proof-net.

Every **MLL** proof-structure is a valid **mCQL** proof-net, with $\otimes$ and $\parr$ identified. The **mCQL** proof-nets so produced never contain loops, so in the following we consider only those **mCQL** proof-nets without loops, without further qualification.

5.3.3 Interpreting MLL in the Free Category

Since, by Theorem 4.26, the proof-nets of **mCQL** represent all the arrows of the free category $F\mathcal{A}$, there is an evident forgetful interpretation of **MLL** in this category: simply derive an **mCQL** proof-net from an **MLL** one by replacing all the occurrences of $\parr$ with $\otimes$. Hence each **MLL** proof-net π determines a map $[\![U\pi]\!] : I \to \bigotimes \Gamma$ of $F\mathcal{A}$, where $U : \mathbf{MLL} \to \mathbf{mCQL}$ is the forgetful map. This map is not faithful: both the proof-nets below have the same translation in **mCQL**.



In the preceding chapter we proved (Lemma 4.20) that an **mCQL** proof-net π over $\mathcal{A}$ is determined by three data: its type Γ, a fixpoint free involution σ over its literals and a functor $\theta : \sigma \to \mathcal{A}$. Given these data, by Lemma 4.19, its denotation in $F\mathcal{A}$ is a map of the following form

$$\begin{aligned}[\![\pi]\!] &= I \xrightarrow{\ulcorner f_1 \urcorner \otimes \cdots \otimes \ulcorner f_n \urcorner} (A_1^* \otimes B_1) \otimes \cdots \otimes (A_n^* \otimes B_n) \xrightarrow{\quad\rho\quad} \Gamma \\ &= I \xrightarrow{\eta_{A_1} \otimes \cdots \otimes \eta_{A_n}} (A_1^* \otimes A_1) \otimes \cdots \otimes (A_n^* \otimes A_n) \xrightarrow{\rho} \Gamma' \xrightarrow{\rho(\overline{f})} \Gamma\end{aligned}$$

where ρ is a permutation derived from σ as described in Lemma 4.21 and $\overline{f} = \otimes_i(1_{A_i^*} \otimes f_i)$. Therefore every **MLL** proof-net has denotation in $F\mathcal{A}$ of the

same form[3]. Since the arrows $\overline{f}$ pertain only to the labelling and not to the underlying proof-net structure we omit them below.

Definition 5.13. Say that a proof-net *has type* $\mathbf{X}$, when its only conclusion is the formula $\mathbf{X}$.

Lemma 5.14. *Let π be an **MLL** proof-net of type $\mathbf{X} * \mathbf{Y}$, where $* \in \{\otimes, \parr\}$, such that $[\![U\pi]\!] = f \otimes g$ for some maps $f : I \to X$ and $g : I \to Y$; then π has type $\mathbf{X} \otimes \mathbf{Y}$.*

Proof. By the above discussion,

$$[\![\pi]\!] = I \xrightarrow{\eta_{A_1} \otimes \cdots \otimes \eta_{A_n}} (A_1^* \otimes A_1) \otimes \cdots \otimes (A_n^* \otimes A_n) \xrightarrow{\rho_1 \otimes \rho_2} X \otimes Y$$

and hence there is no axiom link in π between the literals occurring in $\mathbf{X}$ and those occurring in $\mathbf{Y}$. If the outer connective is $\parr$ then the proof-structure is disconnected in every switching; hence the type is necessarily $\otimes$. □

Lemma 5.15. *Suppose that π has type $\mathbf{X} \otimes \mathbf{Y}$; then $[\![\pi]\!] = [\![\pi_X]\!] \otimes [\![\pi_Y]\!]$.*

Proof. Let $[\![\pi]\!] = \rho \circ (\eta_{A_1} \otimes \cdots \otimes \eta_{A_n})$ as above, and suppose that ρ does not factorise into disjoint permutations upon X and Y; then there is at least one axiom link between the literals of $\mathbf{X}$ and $\mathbf{Y}$. Let x and y be these literals. Since x is a left subformula of the outermost times link, and y is a right subformula, there is a switching connecting each of these to the outer connective, and hence a cycle, contradicting the correctness criterion. □

These lemmas combine to give the following:

Theorem 5.16. *An **MLL** proof-net π is of type $\mathbf{X} \otimes \mathbf{Y}$ if and only if $[\![\pi]\!]$ is separable.*

Therefore, in $F\mathcal{A}$, each point $I \to A \otimes B$ is entangled exactly when it is the denotation of an **MLL** proof-net of type $\mathbf{A} \parr \mathbf{B}$. It is automatic that in $GF\mathcal{A}(\mathbf{I}, \mathbf{A} \otimes \mathbf{B})$ contains $[\![\pi]\!]$ if it is separable; due to the full completeness of $GF\mathcal{A}$ for **MLL** [Tan97] we also have that $[\![\pi]\!] \in GF\mathcal{A}(\mathbf{I}, \mathbf{A} \parr \mathbf{B})$ iff and only if it is entangled.

Theorem 5.17. *Let $[\![\pi]\!] : I \to \Gamma^* \otimes \Delta$ such that*

$$[\![\pi]\!] = \rho \circ (\ulcorner f_1 \urcorner \otimes \cdots \otimes \ulcorner f_n \urcorner)$$

as above. Then $[\![\pi]\!]$ is maximally entangled if and only if all the f_i are unitary and ρ has the property that

$$\rho(A_i^*) \in \Gamma^* \qquad \textit{iff} \qquad \rho(B_i) \in \Delta \tag{$*$}$$

for all i.

Proof. Recall that for $[\![\pi]\!] = \ulcorner g \urcorner$ to be maximally entangled with respect to Γ, Δ, $g : \Gamma \to \Delta$ should be unitary. Explicitly we have

$$g = \Gamma \xrightarrow{1_\Gamma \otimes [\![\pi]\!]} \Gamma \otimes \Gamma^* \otimes \Delta \xrightarrow{\epsilon_\Gamma \otimes 1_\Delta} \Delta,$$

[3]This is indeed a well known fact about **MLL**; see [Gir87b].

which is shown diagrammatically for the case $n = 3$ in Figure 5.3a.

Suppose that $(*)$ does not hold; then there exists i such that: (i) $\rho(A_i^*)$ and $\rho(B_i)$ are both in Γ; or (ii) $\rho(A_i^*)$ and $\rho(B_i)$ are both in Δ. Suppose it is the first. Then $\ulcorner f_i \urcorner$ has its entire codomain in Γ and hence, upto permutation,

$$\begin{aligned} g &= ((\epsilon_{A_i^*} \otimes \epsilon_{B_i}) \circ (1_{A_i} \otimes \ulcorner f_i \urcorner \otimes 1_{B_i^*})) \otimes g' \\ &= \llcorner f_i \lrcorner \otimes g', \end{aligned}$$

and hence

$$g^\dagger \circ g = (\llcorner f_i \lrcorner^\dagger \otimes g'^\dagger) \circ (\llcorner f_i \lrcorner \otimes g')$$

which is always different to 1_Γ regardless of g'. Similarly, if (ii) holds then

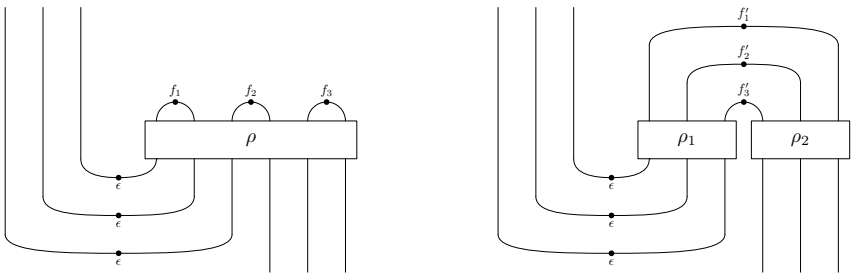
$$g = g'' \otimes \ulcorner f_i \urcorner,$$

(again, upto permutation) from which $g \circ g^\dagger \neq 1_\Delta$. Therefore, if $(*)$ does not hold, then g is not unitary.

Conversely, suppose the condition holds. Note that if ρ sends A_i^* to Δ and B_i to Γ then it factorises as $\rho' \circ \sigma$ where sigma is the permutation $\sigma : A_i^* \otimes B_i \to B_i \otimes A_i^*$, leaving the other elements unchanged. Since $\sigma \circ \ulcorner f_i \urcorner = \ulcorner f_i^* \urcorner$ we can assume without loss of generality that ρ sends the left hand factor of each $\ulcorner f \urcorner$'s codomain to Γ and each right hand factor to Δ. Further $\otimes_i \ulcorner f_i \urcorner$ differs from $\ulcorner f_n \urcorner \circ \ulcorner f_{n-1} \urcorner \circ \cdots \circ \ulcorner f_1 \urcorner$ only by a permutation; we absorb this permutation into ρ. With g now in this form, the condition $(*)$ asserts that ρ factorises into $\rho_1 \otimes \rho_2$, as shown in 5.3b. Further, by choosing the order of the $\ulcorner f_i \urcorner$ we may arrange that $\rho_1 = 1_{\Gamma^*}$. We have then

$$\begin{aligned} g &= \Gamma \xrightarrow{1_\Gamma \otimes \ulcorner f_n \urcorner \circ \cdots \circ \ulcorner f_1 \urcorner} \Gamma \otimes \Gamma^* \otimes \Delta' \xrightarrow{\epsilon_\Gamma \otimes \rho_2} \Delta \\ &= \Gamma \xrightarrow{1_\Gamma \otimes \eta_\Gamma} \Gamma \otimes \Gamma^* \otimes \Delta' \xrightarrow{\epsilon_\Gamma \otimes (\overline{f} \circ \rho_2)} \Delta \\ &= \Gamma \xrightarrow{\overline{f} \circ \rho_2} \Delta. \end{aligned}$$

Hence g is unitary if and only if each of the f_i is unitary. □



(a) (b)

Figure 5.3:

The above theorem may be unpacked to a condition on proof-nets.

Corollary 5.18. *An **MLL** proof-net of type $\mathbf{X} \parr \mathbf{Y}$ codes a maximally entangled state exactly when all its axiom links connect a literal occurring in $\mathbf{X}$ with one occurring in $\mathbf{Y}$, and are labelled by unitaries.*

We have shown that if a point $|\psi\rangle$ in $F\mathcal{A}$ is the denotation of a proof π in **MLL** then the type of π determines whether or not $|\psi\rangle$ is entangled. We now show that, upto a scalar factor, every point of $F\mathcal{A}$ is the denotation of an **MLL** proof.

Theorem 5.19. *Let π be a normal proof-net of **mCQL** which contains no loops and is connected; then there exists a cut-free **MLL** proof-net π' such that $u(\pi') = \pi$.*

Proof. We construct the required proof π' by inductively assigning either $\otimes$ or $\parr$ to each $\otimes$-link of π. Suppose π has n non-axiom links, all of which are $\otimes$-links since π is cut-free. Remove any maximal $\otimes$-link l from the proof-net: if the residual proof-net is connected then l is assigned $\parr$; otherwise it is assigned $\otimes$. Each connected component π_i of the residue determines an **MLL** proof-net by induction. □

The **MLL** typing of an arbitrary **mCQL** proof π is not unique; however, if π is closed then its outer connective is fixed. In other words, if we view the connectives as entanglement descriptors, the entanglement is well defined upto a given partition of the state space; the entanglement within the subspaces is, in general, not well defined. If $\pi \vdash \mathbf{X} \otimes \mathbf{Y}$ then the subproofs corresponding to the subformulae $\mathbf{X}$ and $\mathbf{Y}$ are complete **MLL** proof-nets in their own right and hence they have well defined outer connectives. In this case $[\![\pi]\!]$ is separable, and hence its reduced densities $\rho_{\mathbf{X}}$ and $\rho_{\mathbf{Y}}$ are both pure states. On the other hand, suppose $\pi \vdash \mathbf{X} \parr \mathbf{Y}$, and let π' be the same proof-net, with the outer most $\parr$ link removed. The corresponding **mCQL** proof $u\pi'$ does not admit a unique **MLL** typing. Note that in this situation the reduced densities of $[\![\pi]\!]$ will be mixed states.

The restriction to connected **mCQL** proof-nets is not important since any disconnected **mCQL** proof-net can be connected by additional $\otimes$-links which do not change its denotation. The requirement that π be free of loops is more interesting. Since each loop denotes a scalar, the theorem asserts that each **mCQL** proof can be viewed as an **MLL** proof-net times a scalar, and conversely each **MLL** proof-net fixes a ray among the **mCQL** proof-nets.

Each proof-net's denotation is based on tensor product of unit maps:

$$I \xrightarrow{\eta\otimes\cdots\otimes\eta} (A^* \otimes A) \otimes \cdots \otimes (A^* \otimes A) \longrightarrow \cdots$$

As a result each proof-net is interpreted as a number of maximally entangled "Bell pairs" which are distributed between the two subsystems. To handle more general multipartite entanglement the generating category must bear a non-trivial monoidal structure. This is carried out in the next chapter. However the bipartite case is of considerable practical interest and so merits separate treatment.

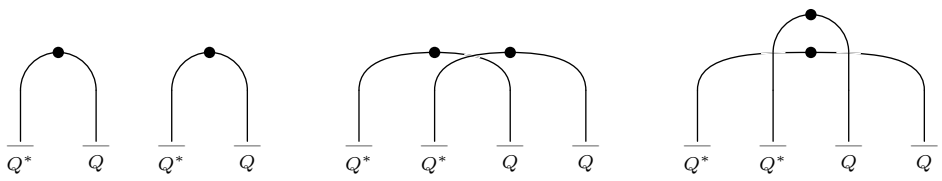
Remark. While this presentation of multiplicative linear logic used the traditional binary formulation of the connectives, there is no essential obstacle to using their n-ary variants, cf. [DR89]. Binary connectives have the advantage that, when viewing points as names of maps, there is a clear domain and codomain.

Chapter 6

Generalised Multiplicative Categorical Quantum Logic

The formulation of **mCQL** in Chapter 4 allows quantum states to be represented as proof-nets whose axioms are taken from a category of unitary maps. However, as the preceding chapter showed, the class of states thus representable is limited to separable states and maximally entangled bipartite states. This class is insufficient for general quantum computation.

The root of this lack of expressivity is the choice to generate the syntax from a bare category; a non-trivial tensor structure is required. Consider a 4-qubit state, ϕ. If ϕ has a proof-net representative π then necessarily its entanglement graph is one of



and hence, it must decompose into two entangled pairs. To represent a genuine 4-way entanglement between the qubits, axioms which do not decompose in this fashion are required.

In any compact closed category the duality between points and maps means that the structure of the entangled states is fixed by the structure of the maps in the category. In the categories we used in the preceding chapters the tensor product was freely constructed, and hence each arrow expresses a relationship between exactly two atomic objects: every arrow in $F\mathcal{A}$ is decomposable into several primitive factors. This limits the possible entanglement between atomic objects to the bipartite case.

An obvious approach to this problem is to build a free compact closed category on a category $\mathcal{A}$ equipped with its own tensor structure, such that the free tensor product of $F\mathcal{A}$ coheres with that of $\mathcal{A}$. In this setting the entangled states would correspond to morphisms which do not decompose as $h = f \otimes g$.

However this approach yields two inequivalent means of introducing a tensor product into a proof-net: in the usual way as a connective, or via an axiom. These correpond to the freely generated tensor and the tensor of the generating

category $\mathcal{A}$ respectively. This duplication is not merely mathematically inelegant, it is an impediment to any study of entanglement since only former of these is amenable to analysis at the level of proof-nets. Lacking any natural way to expose internal tensor of the axiom scheme in the proof-net setting we instead seek an axiom scheme which does not internalise the tensor structure; we still want each arrow of $\mathcal{A}$ to represent an inseparable state. This requirement leads naturally to the approach followed in this chapter, that of constructing the free compact closed category generated by a *compact symmetric polycategory*.

The notion of polycategory is a strict generalisation of category; each polyarrow has a list of objects for its domain and codomain, rather than a single object. Polycategories have been previously studied in the context of classical logic [Lam69, Sza75], rewriting grammars [Vel88] and bicategories [Lam05]. Blute, Ivanov and Panangaden have used polycategories to study quantum causal systems [BIP03], though their approach is quite different to that pursed here. The most important result relating polycategories to monoidal categories is by Cockett and Seely [CS97] who showed that the essential structure of a polycategory is that of a linearly distributive category, which itself is essentially a model of multiplicative linear logic without negation.

In Section 6.1 we present the axioms for compact polycategories. As in the case of **mCQL** proof-nets, operating in the compact setting eliminates many difficulties concerned with sequentialisability present in the standard theory of polycategories; see for example [Kos03], Section 1. The compact definition of polycategory produces a structure extremely close to a symmetric monoidal category; the principal difference is that a polycategory permits parallel composition only under the scope of a sequential composition, hence there are no "non-interacting" composites as there are in a symmetric monoidal category. In a sense, a compact polycategory is a monoidal category without a tensor.

The main purpose of this chapter is to construct the free compact closed category generated by a polycategory, interpreting multi-ary polyarrows as arrows between tensor products. To do so we need to generalise the work of Kelly and Laplaza in [KL80]. Recall that the morphisms of the free compact closed category upon a category $\mathcal{A}$ consist of two kinds of structures: arcs labelled by arrows of $\mathcal{A}$ and loops labelled by equivalence classes of endomorphisms in $\mathcal{A}$. The loops can be understood as the result of tracing out the entire domain and codomain of an arrow. Hence each element of the structure is either an arrow of $\mathcal{A}$ or an equivalence class of the same. When the free construction is performed over a polycategory the situation is more complicated since it is possible to take a partial trace, leaving behind a non-trivial morphism which is not a mere combination of arrows of the generator. As a result the structure of morphisms may be arbitrary connected graphs, with the trace providing cycles.

In Sections 6.2 and 6.3 we introduce a class of circuit diagrams to represent the free compact closed structure constructed upon a polycategory, and prove that the category of such diagrams does indeed have the desired universal property. In fact, for a given compact symmetric polycategory $\mathcal{P}$, we have a sequence of embeddings

$$\mathcal{P} \hookrightarrow \mathrm{SM}(\mathcal{P}) \hookrightarrow \mathrm{TR}(\mathcal{P}) \hookrightarrow \mathrm{CC}(\mathcal{P})$$

into the free strict symmetric monoidal, symmetric traced monoidal, and compact closed categories generated by $\mathcal{P}$. Each of these can be identified with a

certain subclass of diagrams, generated from the polycategory by adding extra objects and arrows, so that the category is closed under the mix rule[1], trace and polarities, respectively[2]. The structure of the commutative monoid of scalars is examined in Section 6.4.

In Section 6.3 we construct the free compact closed category generated by a polycategory. However, the construction is not completely general: the polycategory must itself be freely generated from a set of basic polyarrows. This restriction is removed in Section 6.5, where we introduce the notion of *homotopy*, a topological equivalence relation. Homotopy for circuits allows the composition and symmetry of the polycategory to be internalised, so circuits which are homotopy equivalent contain the same elements of the underlying polycategory, regardless of how these elements are presented. The homotopy relation does not affect the additional structure introduced by the free construction — namely mix, polarities and trace — and hence every connected component of the diagram is, up to polarity, a trace equivalence class of polyarrows. This completes the description of the free compact closed category generated by a polycategory.

Having established a suitable generalisation of Kelly and Laplaza's coherence theorem we turn our attention to a logical formulation of this structure. In Section 6.6 we generalise the proof-nets for **mCQL** in two ways. Firstly, we switch to two-sided proof-nets, in order to represent processes as well as states. Secondly, the axiom links are multi-ary links drawn from a polycategory $\mathcal{P}$. The resulting system is similar to those of [BCST96, Tan97], but, as is well known [Tak87] ,the presence of non-logical axioms is an obstacle to cut elimination. Blute has considered **MLL** with non-logical axioms [Blu93]; the system he used however requires that each axiom be equipped with a Kelly-Mac Lane graph to enable cut elimination. This course of action would defeat our present purpose: if every 4-ary axiom can be decomposed into a pair of binary links then once again only bipartite entanglements are possible.

So we press on without a true cut-elimination theorem. However, all is not lost: generalised proof-nets enjoy strong normalisation and the normal form of each net consists of a polyarrow wrapped in logical connectives. This characterisation of the normal forms leads naturally to the result that the proof-net syntax gives a fully-complete representation of the free category.

As we noted in Chapter 3, it is often necessary to impose equations on a free model in order to obtain a more accurate representation of the properties of realistic quantum systems. Section 6.7 considers the situation where equations between the generators are given as an explicit rewrite system. We show that the rewriting on $\mathcal{P}$ can be lifted to the circuit formalism, and hence also to the proof-net notation; further, if the original rewrite system was confluent, then so is the lifted version. The same does not hold for termination.

The final section is a simple example. The circuit formalism is used to provide an abstract proof of the no-cloning theorem.

---

[1] There are several equivalent ways to get a symmetric monoidal category from a polycategory: introduce the mix rule; allow nullary composition, or introduce identity polyarrows for all sequences of objects.

[2] This sequence of constructions is carried out in setting where the generator is a category, rather than a polycategory in [Abr05]

6.1 Polycategories

Polycategories were introduced by Lambek [Lam69] (see also Szabo [Sza75] for a more formal account) with the intention of providing a categorical framework for classical logic, with multiple formulae on both left and right of the sequent. Their composition law is based on the cut rule of classical logic. We generalise this definition by permitting arrows to be composed along many premises at the same time, known as *multi-cut*.

Definition 6.1. A *compact symmetric polycategory* (or *symmetric polycategory with multicut*), $\mathcal{P}$, consists of a class of objects $\mathrm{Obj}_{\mathcal{P}}$ and, to each pair (Γ, Δ) of finite sequences over $\mathrm{Obj}_{\mathcal{P}}$, a set of polyarrows $\mathcal{P}(\Gamma, \Delta)$. Given a non-empty sequence of objects Θ and poly-arrows

$$\Gamma \xrightarrow{f} \Delta_1, \Theta, \Delta_2 \qquad \text{and} \qquad \Gamma_1, \Theta, \Gamma_2 \xrightarrow{g} \Delta$$

we may form the composition

$$\Gamma_1, \Gamma, \Gamma_2 \xrightarrow{g \,{}_i\overset{k}{\circ}_j f} \Delta_1, \Delta, \Delta_2$$

where $|\Delta_1| = i$, $|\Gamma_1| = j$ and $|\Theta| = k > 0$. In general there may be many ways to compose a pair of polyarrows. It is helpful to visualise composition in polycategories in terms of "wiring" diagrams; in this case, composition would be represented as shown in Figure 6.1. If A is in $\mathrm{Obj}_{\mathcal{P}}$ then there is an identity

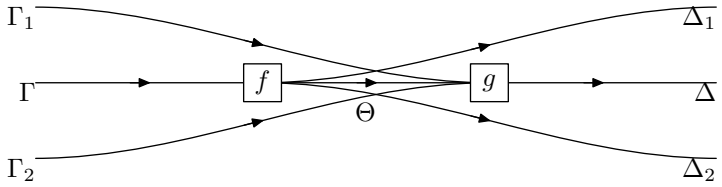


Figure 6.1: Composition of polyarrows

arrow 1_A from the singleton sequence A to itself, such that

$$f \circ 1_A = f \qquad \text{and} \qquad 1_A \circ g = g$$

where the ith object of $\mathrm{dom}\, f$ and $\mathrm{cod}\, g$ is A.

Composition is associative: given three polyarrows

$$\begin{aligned} \Gamma &\xrightarrow{f} \Delta_1, \Theta, \Delta_2, \\ \Gamma_1, \Theta, \Gamma_2 &\xrightarrow{g} \Delta_3, \Phi, \Delta_4, \\ \Gamma_3, \Phi, \Gamma_4 &\xrightarrow{h} \Delta \end{aligned}$$

then

$$h \,{}_{(i+i')}\overset{k'}{\circ}_{j'} (g \,{}_i\overset{k}{\circ}_j f) = (h \,{}_{i'}\overset{k'}{\circ}_{j'} g) \,{}_i\overset{k}{\circ}_{(j+j')} f, \tag{6.1}$$

where $i' = |\Delta_3|$, $j' = |\Gamma_3|$, $k' = |\Phi|$, and with i, j, k as above. Diagrammatically this asserts that the composition shown in Figure 6.2 is unambiguous. We shall omit the indices on polycomposition wherever it is unambiguous to do so.

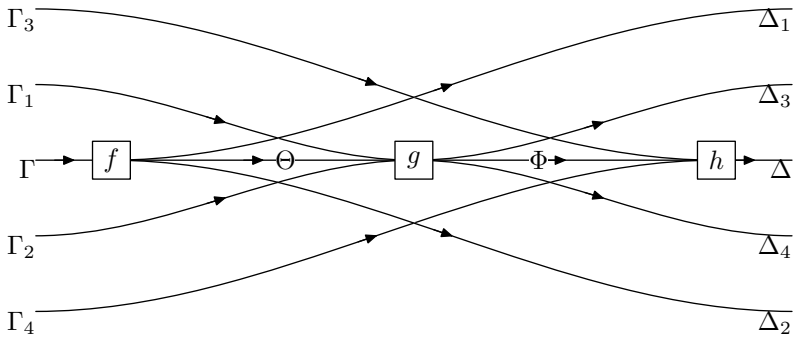


Figure 6.2: Associativity of poly-composition

Suppose that σ is a permutation on the objects of Γ, and τ is a permutation on Δ; then for every $f \in \mathcal{P}(\Gamma, \Delta)$ there is a poly-arrow $f^\tau_\sigma \in \mathcal{P}(\sigma\Gamma, \tau\Delta)$. Such permutations compose in the obvious fashion,

$$(f^\tau_\sigma)^{\tau'}_{\sigma'} = f^{\tau'\tau}_{\sigma'\sigma}.$$

Given the poly-arrows $f : \Gamma \to \Delta_1, \Theta, \Delta_2$ and $g : \Gamma_1, \Theta, \Gamma_2 \to \Delta$, and permutations $\sigma_1, \sigma_2, \tau_1, \tau_2, \rho$ such that σ_1 acts only on Γ_1, Γ_2, σ_2 only on Γ, τ_1 only on Δ_1, Δ_2, τ_2 only Δ and ρ only on Θ, then the following coherence conditions apply:

$$(g_{\sigma_1}) \circ f = (g \circ f)_{\sigma_1}; \tag{6.2}$$
$$g \circ (f_{\sigma_2}) = (g \circ f)_{\sigma_2}; \tag{6.3}$$
$$g \circ (f^{\tau_1}) = (g \circ f)^{\tau_1}; \tag{6.4}$$
$$(g^{\tau_2}) \circ f = (g \circ f)^{\tau_2}; \tag{6.5}$$
$$g \circ (f^\rho) = (g_{\rho^{-1}}) \circ f. \tag{6.6}$$

Aside from associativity, identities, and symmetry, we also require that composition satisfies two commutativity conditions. Given

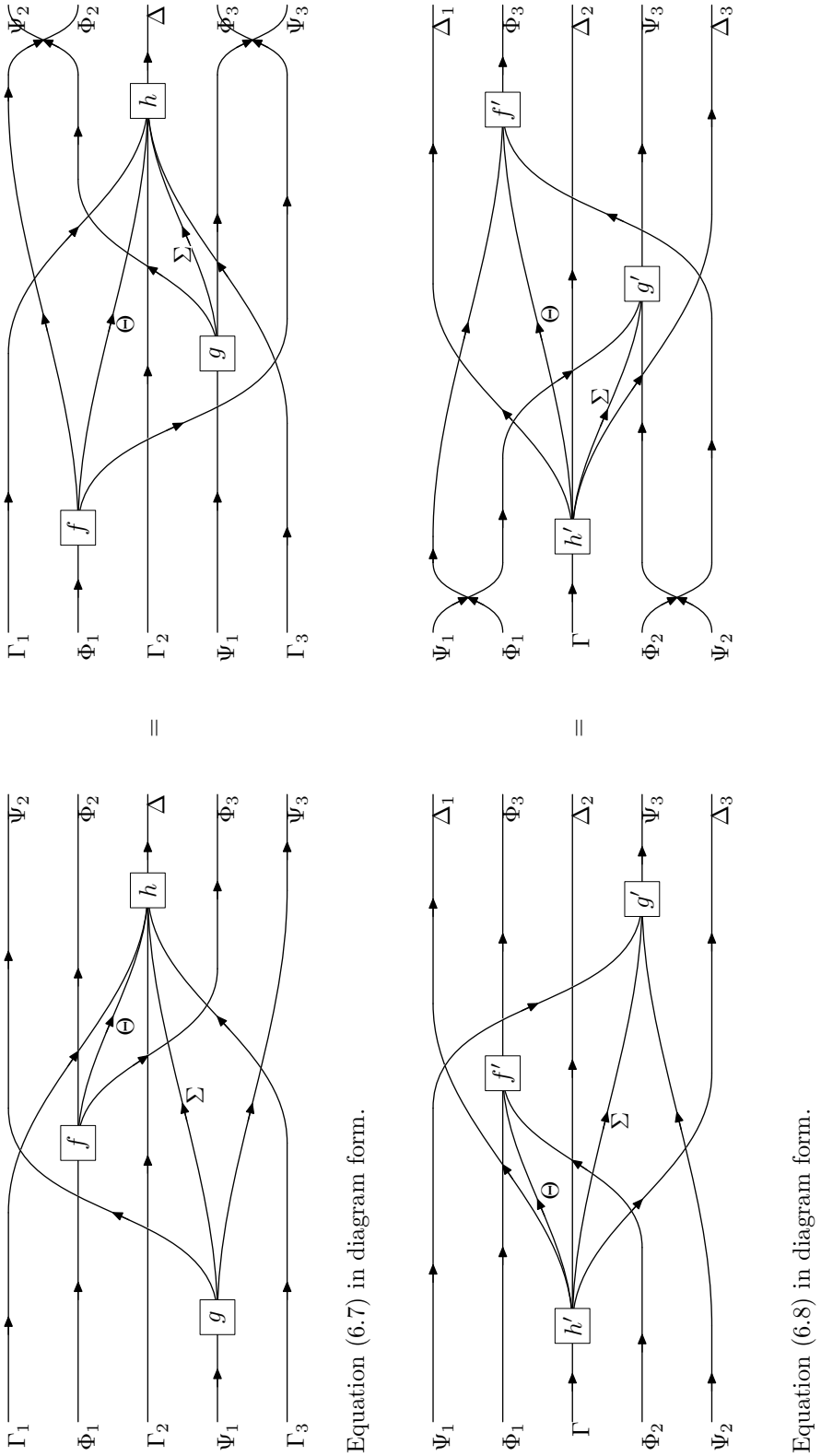
$$\Phi_1 \xrightarrow{f} \Phi_2, \Theta, \Phi_3 \qquad \Phi_1, \Theta, \Phi_2 \xrightarrow{f'} \Phi_3$$
$$\Psi_1 \xrightarrow{g} \Psi_2, \Sigma, \Psi_3 \qquad \Psi_1, \Sigma, \Psi_2 \xrightarrow{g'} \Psi_3$$
$$\Gamma_1, \Theta, \Gamma_2, \Sigma, \Gamma_3 \xrightarrow{h} \Delta \qquad \Gamma \xrightarrow{h'} \Delta_1, \Theta, \Delta_2, \Sigma, \Delta_3$$

let τ be the permutation sending $\Phi_2, \Psi_2, \Delta, \Psi_3, \Phi_3$ to $\Psi_2, \Phi_2, \Delta, \Phi_3, \Psi_3$ and σ be that sending $\Psi_1, \Phi_1, \Gamma, \Phi_2, \Psi_2$ to $\Phi_1, \Psi_1, \Gamma, \Psi_2, \Phi_2$. The following must hold:

$$(h \circ f) \circ g = ((h \circ g) \circ f)^\tau, \tag{6.7}$$
$$f' \circ (g' \circ h') = (g' \circ (f' \circ h'))_\sigma. \tag{6.8}$$

Figure 6.3 presents these equations diagrammatically. Finally one further axiom, for generalised associativity, is required. Given polyarrows

$$f : \Gamma \longrightarrow \Delta_1, \Psi_1, \Phi, \Psi_2, \Delta_2$$
$$g : \Gamma_1, \Phi, \Gamma_2 \longrightarrow \Delta_3, \Theta, \Delta_4$$
$$h : \Gamma_3, \Psi_1, \Theta, \Psi_2, \Gamma_4 \longrightarrow \Delta$$



Equation (6.7) in diagram form.

Equation (6.8) in diagram form.

Figure 6.3: Commutativity Conditions

and permutations

$$\sigma : \Delta_1, \Psi_1, \Delta_3, \Theta, \Delta_4, \Psi_2, \Delta_2 \longmapsto \Delta_1, \Delta_3, \Psi_1, \Theta, \Psi_2, \Delta_4, \Delta_2$$
$$\tau : \Gamma_3, \Gamma_1, \Psi_1, \Phi, \Psi_2, \Gamma_2, \Gamma_4 \longmapsto \Gamma_3, \Psi_1, \Gamma_1, \Phi, \Gamma_2, \Psi_2, \Gamma_4$$

then

$$h \circ (g \circ f)^\tau = (h \circ g)_\sigma \circ f. \tag{6.9}$$

The generalised associativity axiom makes the diagram shown in Figure 6.4 unambiguous. In this situation Ψ_1 and Ψ_2 may be empty; if both are empty

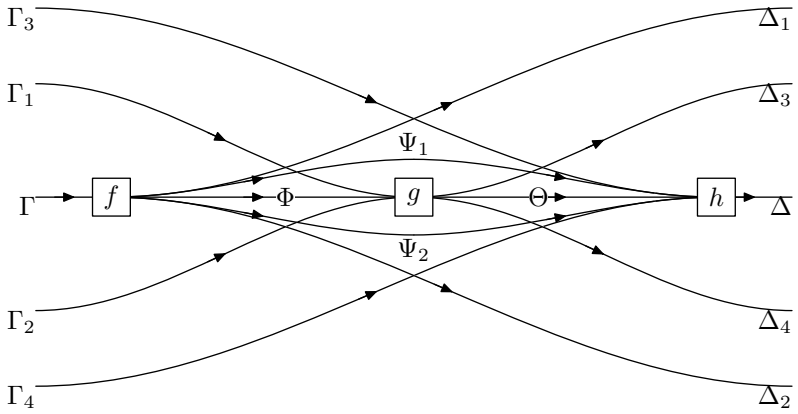


Figure 6.4: Generalised Associativity Axiom

then this axiom boils down to straight associativity.

If the composition $f \,{}_i\!\overset{k}{\circ}_j\, g$ is defined only in the case where $k = 1$ then the definition above gives a (non-compact) symmetric polycategory in the usual sense.

Henceforth all polycategories, with or without multicut, are assumed to be symmetric.

Given a map F and a list of objects $\Gamma = A_1, A_2, \ldots, A_n$, we write $F\Gamma$ for the list $FA_1, FA_2, \ldots, FA_n$.

Definition 6.2. A *functor* $F : \mathcal{A} \to \mathcal{B}$ between polycategories is a map $F_O : \mathrm{Obj}_\mathcal{P} \to \mathrm{Obj}_\mathcal{Q}$ and a family $F_{\Gamma,\Delta} : \mathcal{P}(\Gamma, \Delta) \to \mathcal{Q}(F_O\Gamma, F_O\Delta)$ such that identities and composition are preserved, as are Equations. (6.1)–(6.9).

Definition 6.3. A natural transformation $\tau : F \Rightarrow G$ between polycategory functors is a family of polyarrows $\tau_\Gamma : F\Gamma \to G\Gamma$ such that

$$\begin{array}{ccc} F\Gamma & \xrightarrow{\tau_\Gamma} & G\Gamma \\ {\scriptstyle Ff}\downarrow & & \downarrow{\scriptstyle Gf} \\ F\Delta & \xrightarrow[\tau_\Delta]{} & G\Delta \end{array}$$

commutes. If there exists another natural transformation $\sigma : G \Rightarrow F$ such that, for all singletons A, $\sigma_A \circ \tau_A = 1_{FA}$ and $\tau_A \circ \sigma_A = 1_{GA}$ then τ is a natural isomorphism.

Every category is a polycategory; a polycategory functor whose domain is a category is just a functor in the usual sense. However if a polycategory $\mathcal{P}$ has any polyarrows whose domain or codomain are not singletons, then there is no functor from $\mathcal{P}$ to any category $\mathcal{C}$, since $\mathcal{C}$ has no polyarrows. However, if $\mathcal{C}$ is equipped with a strict symmetric tensor, a useful notion of functor arises immediately.

Definition 6.4. A symmetric monoidal functor F from a polycategory $\mathcal{P}$ to a symmetric monoidal category $\mathcal{C}$ consists of a an object map $F_O : \mathrm{Obj}_{\mathcal{P}} \to \mathrm{Obj}_{\mathcal{C}}$ and an arrow map F_A sending each hom-set $\mathcal{P}(\Gamma, \Delta)$ to $\mathcal{C}(\bigotimes F_O\Gamma, \bigotimes F_O\Delta)$ such that, for all $\Gamma \xrightarrow{f} \Delta_1, \Theta, \Delta_2$ and $\Gamma_1, \Theta, \Gamma_2 \xrightarrow{g} \Delta$,

$$\begin{aligned} F(g \circ f) &= (1_{F\Delta_1} \otimes Fg \otimes 1_{F\Delta_2}) \circ \sigma \circ (1_{F\Gamma_1} \otimes Ff \otimes 1_{F\Gamma_2}), \\ F(f^{\tau_1}_{\tau_2}) &= \tau_1 \circ Ff \circ \tau_2^{-1}, \end{aligned}$$

where σ is the permutation exchanging Δ_i and Γ_i, while leaving Θ unchanged.

It is easy to verify that this definition preserves the coherence conditions above. Natural transformations between such functors are simply natural transformations in the usual sense.

It is often convenient to work with formal polycategories which are freely generated from some sets of objects and polyarrows. For example, let $\mathcal{Q}$ be the polycategory whose only object is Q, and which is generated by the following non-identity polyarrows:

$$\begin{aligned} |0\rangle, |1\rangle &: - \to Q \\ \langle 0|, \langle 1| &: Q \to - \\ H, X, Y, Z &: Q \to Q \\ E &: Q, Q \to Q, Q \end{aligned}$$

We can now define a symmetric monoidal functor $[\![\cdot]\!] : \mathcal{Q} \to \mathbf{Vect}^{\mathrm{fd}}_{\mathbb{C}}$ by assigning

$$[\![Q]\!] = \mathbb{C}^2$$

and

$$[\![|0\rangle]\!] = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \qquad [\![|1\rangle]\!] = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \qquad [\![\langle 0|]\!] = \begin{pmatrix} 1 & 0 \end{pmatrix}, \qquad [\![\langle 1|]\!] = \begin{pmatrix} 0 & 1 \end{pmatrix},$$

$$[\![H]\!] = \tfrac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \quad [\![X]\!] = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad [\![Y]\!] = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \quad [\![Z]\!] = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix},$$

$$[\![E]\!] = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}.$$

In this example, $[\![\cdot]\!]$ defines the standard interpretation of $\mathcal{Q}$ in the category of vector spaces.

6.2 Graphs and Circuits

We introduce a class of formal *circuits*. These provide a type-free description of the compact closed structure constructed atop a polycategory. While they are based on the generalised graphs of [JS91, JSV96], we neglect the geometric and topological issues given prominence there in favour of a more combinatorial approach.

6.2.1 Graphs

Definition 6.5. A *directed graph* consists of a 4-tuple (V, E, s, t) where V and E are sets, respectively of *vertices* and *edges*, s and t are maps

$$E \underset{t}{\overset{s}{\rightrightarrows}} V$$

which we call *source* and *target*. Let $\mathrm{in}(v)$ and $\mathrm{out}(v)$ be V-indexed subsets of E defined by

$$\begin{aligned} \mathrm{in}(v) &= t^{-1}(v) \\ \mathrm{out}(v) &= s^{-1}(v). \end{aligned}$$

The *in-degree* of a vertex v is the cardinality of $\mathrm{in}(v)$ and the *out-degree* is the cardinality of $\mathrm{out}(v)$. The *degree* of a vertex is the sum of its in- and out-degrees.

Now we define some standard terminology. Let G be a directed graph. If there exists an edge e such that $s(e) = x$ and $t(e) = y$ then write $x \to y$; we say that y is said to be a *successor* of x and x is a *predecessor* of y. If either $x \to y$ or $y \to x$ then x and y are *adjacent*. For vertices x, y we write $x \prec y$ if there exists a non-empty finite sequence of edges $\{e_i\}_{i=0}^{n}$ such that

$$\begin{aligned} & s(e_0) = x, \\ & t(e_n) = y, \\ \forall i \quad & s(e_{i+1}) = t(e_i). \end{aligned}$$

Such a sequence of edges is called a *directed path* from x to y. A pair of vertices x and y are said to be *connected* if x and y are related by the symmetric transitive closure of $\prec$. The graph G is said to be connected if every pair of vertices is connected; G is *acyclic* if, for all vertices x and y, $x \prec y$ implies $y \not\prec x$. A subset of V is a *connected component* of G if every pair of vertices in V are connected and no vertex in V is connected to a vertex outside V.

Lemma 6.6. *Let $G = (V, E, s, t)$ be an acyclic directed graph and let $\sqsubseteq$ be the reflexive closure of $\prec$; then $\sqsubseteq$ is a partial order on V.*

Proof. The relation is reflexive by definition so we show transitivity and anti-symmetry.

Let x, y, z be vertices such that $x \sqsubseteq y$ and $y \sqsubseteq z$. Let $(e_i)_i$ and $(f_j)_j$ be the possibly empty paths connecting x to y and y to z; then $e_0, \ldots, e_m, f_0, \ldots, f_n$ is a path from x to z, hence $x \sqsubseteq z$.

If $x \sqsubseteq y$ and $y \sqsubseteq x$ then either $x = y$ or both $x \prec y$ and $y \prec x$; but G is assumed to be acyclic, so the second possibility is excluded. □

Corollary 6.7. *Immediately from the above:*

1. *if G is acyclic the relation $\prec$ is a strict partial order on V;*
2. *if G contains a cycle then $\sqsubseteq$ is a preorder on V.*

Definition 6.8. A *directed graph with circles* is a 5-tuple (V, E, C, s, t) such that (V, E, s, t) is a directed graph and C is a set of *circles*. The set of edges of a graph with circles is the disjoint union $E + C$.

Of course, graphs are naturally drawn as pictures. Figure 6.5 (a) shows a pictorial representation of the kinds of edges permitted in a directed graph with circles; 6.5 (b) shows examples which are not permitted by this definition.

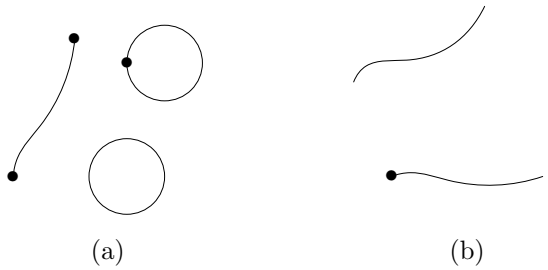


Figure 6.5: (a) Permitted and (b) forbidden edges

A directed graph with circles is called acyclic if its underlying directed graph is acyclic and $C = \emptyset$. Each circle is considered to be a connected component hence a graph with circles is connected if (V, E, s, t) is connected and $C = \emptyset$ or if $V = E = \emptyset$ and C is a singleton.

We now define a pair of mutually inverse operations for "splitting" and "fusing" edges of a graph.

Definition 6.9. Let Γ be a graph with circles, and let e be an edge in Γ. Let $\Gamma_{\lhd e}$ to the result of *splitting* e; precisely:

$$\Gamma_{\lhd e} = (V + \{e\},\ (E - \{e\}) + \{e_1, e_2\},\ C, s', t')$$

where e_1 and e_2 are fresh edges. Define

$$\begin{array}{ll} s'(e_1) = s(e), & t'(e_1) = e, \\ s'(e_2) = e, & t'(e_2) = t(e), \end{array}$$

and let s' and t' otherwise agree with s and t.

Remark. In this section we denote the disjoint union of two sets as $A + B$. To avoid getting mired in relabelling, this operation is taken as strictly associative. Further, given distinct sets A and B, I will assume that $A + B = B + A$, unless it is ambiguous to do so. As a consequence, some results stated as equalities are properly natural isomorphisms; this loss of precision is harmless since in later sections we will always consider graphs up to isomorphism. In return, considerable notational baggage may be dropped.

Lemma 6.10. *In $\Gamma_{\lhd e}$, e is a vertex with in-degree 1 and out-degree 1.*

Proof. Immediate from the definition. □

Lemma 6.11. *Let e, f be distinct vertices in Γ; then $(\Gamma_{\lhd e})_{\lhd f} = (\Gamma_{\lhd f})_{\lhd e}$.*

Proof. Let $(\Gamma_{\lhd e})_{\lhd f} = (V_1, E_1, C, s_1, t_1)$ and $(\Gamma_{\lhd f})_{\lhd e} = (V_2, E_2, C, s_2, t_2)$. Then

$$V_1 = V + \{e\} + \{f\} = V + \{f\} + \{e\} = V_2$$

and

$$\begin{aligned} E_1 &= (((E - \{e\}) + \{e_1, e_2\}) - \{f\}) + \{f_1, f_2\} \\ &= (E - \{e, f\}) + \{e_1, e_2\} + \{f_1, f_2\} \\ &= (((E - \{f\}) + \{f_1, f_2\}) - \{e\}) + \{e_1, e_2\} \\ &= E_2. \end{aligned}$$

Let

$$s^* = \lambda xyz \begin{cases} \text{if } z = x_1 \text{ then } s(x) \\ \text{else if } z = x_2 \text{ then } x \\ \text{else} \begin{cases} \text{if } z = y_1 \text{ then } s(y) \\ \text{else if } z = y_2 \text{ then } y \\ \text{else } s(z) \end{cases} \end{cases}$$

then $s_1 = s^*fe$ and $s_2 = s^*ef$; but since e and f are distinct, the inner and outer conditionals of s^* may commute, hence $s_1 = s_2$. The target maps t_1 and t_2 are treated similarly. □

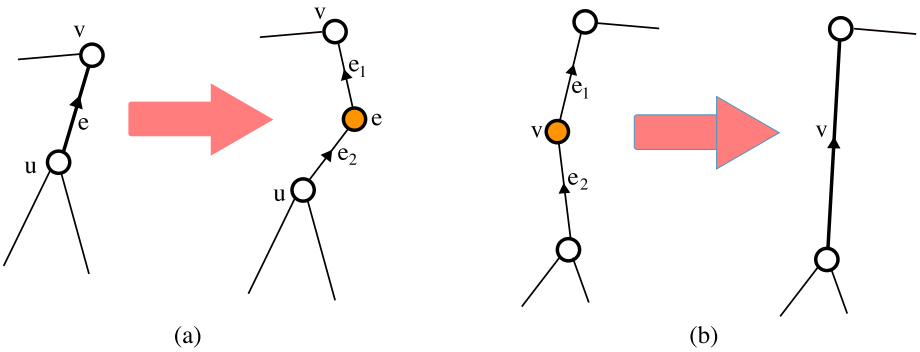


Figure 6.6: (a) Splitting an edge; (b) Fusing two edges

In a graph with circles, the circles are considered to be honourary edges. We can define splitting for circles too: suppose that $c \in C$, then

$$\Gamma_{\lhd c} = (V + \{c\},\ E + \{c_1, c_2\},\ C - \{c\}, s', t')$$

where s', t' are as defined above. By the same reasoning employed in lemma 6.11 above, splitting distinct circles is commutative, and also commutes with edge-splitting. Henceforth no distinction will be made between edge- and circle-splitting.

Corollary 6.12. *Let $F \subset E + C$; then $\Gamma_{\lhd F}$ is well defined.*

We now define the inverse operation to edge splitting, called edge fusion. Suppose that v is a vertex of $\Gamma = (V, E, C, s, t)$ such that $|\text{in}(v)| = |\text{out}(v)| = 1$, and suppose that e_1 and e_2 are its incident edges:

$$\cdots \xrightarrow{e_1} v \xrightarrow{e_2} \cdots .$$

Definition 6.13. With Γ, v, e_1 and e_2 as above, define $\Gamma_{\triangleright v}$, the *fusing of edges at* v, to be the graph obtained by removing v and identifying e_1 and e_2. There are two cases:

1. $e_1 \neq e_2$: Since v has degree 2, this implies that $s(e_1) \neq v$ and $t(e_2) \neq v$; the two edges are merged. Define:

   $$\Gamma_{\triangleright} v = (V - \{v\},\ (E - \{e_1, e_2\}) + \{v\},\ C, s', t')$$

   and

   $$s'(e) = \begin{cases} s(e) & \text{if } e \neq v \\ s(e_1) & \text{if } e = v \end{cases}$$

   $$t'(e) = \begin{cases} t(e) & \text{if } e \neq v \\ t(e_2) & \text{if } e = v \end{cases}$$

2. $e_1 = e_2$: The edge e_1 has both source and target at v, and there are no other edges incident at v; removing the vertex leaves a circle. Define:

   $$\Gamma_{\triangleright} v = (V - \{v\}, E - \{e_1\}, C + \{v\}, s', t')$$

   where s' and t' are the respective restrictions of s and t to $E - \{e_1\}$.

Lemma 6.14. *Let x, y be vertices of Γ such that both x and y have in- and out-degree equal to 1; then $(\Gamma_{\triangleright x})_{\triangleright y} = (\Gamma_{\triangleright y})_{\triangleright x}$.*

Proof. Omitted, but essentially similar to Lemma 6.11. □

Corollary 6.15. *Let W be a set of vertices all with in-degree and out-degree equal to 1; then the operation $\Gamma_{\triangleright W}$ is well defined.*

Lemma 6.16. *Let Γ be a graph containing an edge e and a vertex v of in- and out-degree 1:*

- $\Gamma = (\Gamma_{\triangleleft e})_{\triangleright e}$;
- $\Gamma = (\Gamma_{\triangleright v})_{\triangleleft v}$.

Proof. Immediate from the definitions. □

Definition 6.17. A *graph with boundary* is a pair $(G, \partial G)$ of an underlying directed graph $G = (V, E, C, s, t)$ and a distinguished subset of the degree one vertices ∂G called the *boundary* of G; $V - \partial G$ is called the *interior* of G, written I_G. If a vertex $x \in \partial G$ it is an *outer* or *boundary node*; otherwise it is an *inner* or *interior node*.

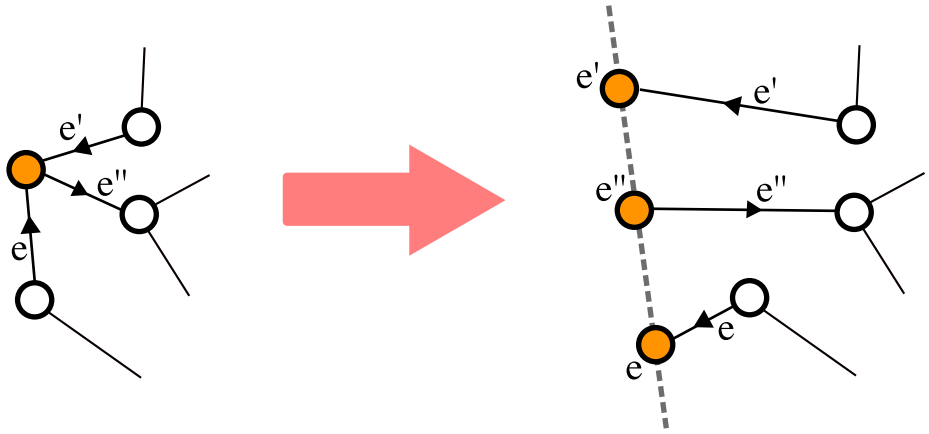


Figure 6.7: An example of breaking a vertex.

The operations of splitting and fusing lift automatically to graphs with boundary. Since all the vertices involved have degree 2 they can never be boundary nodes, therefore if Γ is a graph with boundary define $\partial(\Gamma_{\triangleright v}) = \partial\Gamma$ and $\partial(\Gamma_{\triangleleft e}) = \partial\Gamma$. We now define a third operation, which we call *breaking* a vertex:

Definition 6.18. Suppose that $\Gamma = (G, \partial G)$ is a graph with boundary and suppose that v is vertex of G. We define a new graph with boundary $\Gamma - v = (G', \partial G')$ by

$$G' = ((V - \{v\}) + \mathrm{in}(v) + \mathrm{out}(v), E, C, s', t')$$

where

$$\begin{aligned} s'(e) &= e \qquad \text{if } e \in \mathrm{out}(v) \\ s'(e) &= s(e) \qquad \text{otherwise} \end{aligned}$$

and

$$\begin{aligned} t'(e) &= e \qquad \text{if } e \in \mathrm{in}(v) \\ t'(e) &= t(e) \qquad \text{otherwise .} \end{aligned}$$

The new boundary is defined

$$\partial G' = (\partial G - \{v\}) + \mathrm{in}(v) + \mathrm{out}(v).$$

It is not necessary that $\mathrm{in}(v)$ and $\mathrm{out}(v)$ be disjoint; hence in general the elements of V' must be tagged to indicate which part of the disjoint union they came from.

Lemma 6.19. *Let x, y be distinct vertices in Γ. Neglecting the coproduct injections,*

$$(\Gamma - x) - y = (\Gamma - y) - x.$$

Proof. Let $\Gamma_1 = (\Gamma - x) - y$ and $\Gamma_2 = (\Gamma - y) - x$, and let V_1, V_2 be their respective sets of vertices. Evidently $\{x, y\}$ is disjoint from each of $\mathrm{in}(x), \mathrm{in}(y), \mathrm{out}(x)$ and $\mathrm{out}(y)$, hence:

$$\begin{aligned} V_1 &= (((V - \{x\}) + \mathrm{in}(x) + \mathrm{out}(x)) - \{y\}) + \mathrm{in}(y) + \mathrm{out}(y) \\ &= (((V - \{x, y\}) + \mathrm{in}(x) + \mathrm{in}(y) + \mathrm{out}(x) + \mathrm{out}(y) \\ &= (((V - \{y\}) + \mathrm{in}(y) + \mathrm{out}(y)) - \{x\}) + \mathrm{in}(x) + \mathrm{out}(x) \\ &= V_2. \end{aligned}$$

The same reasoning, replacing V with $\partial\Gamma$, shows that the boundaries also coincide. It is possible that an edge of Γ be incident at both x and y, in which case it will occur twice in $V_1 (= V_2)$; in this case we use subscripts x, y to indicate which occurrence is intended. Define a map

$$\mathrm{src} = \lambda v_1 v_2 \,.\, e \mapsto \begin{cases} e_{v_2} \text{ if } e \in \mathrm{out}(v_2) \\ \text{else } \begin{cases} e_{v_1} \text{ if } e \in \mathrm{out}(v_1) \\ \text{else } s(e) \end{cases} \end{cases},$$

then $s_1 = \mathrm{src}\, x\, y$ and $s_2 = \mathrm{src}\, y\, x$. However, since $\mathrm{out}(x)$ and $\mathrm{out}(y)$ are disjoint, these two maps coincide. The target maps are treated in the same fashion, hence $s_1 = s_2$ and $t_1 = t_2$, making the two graphs equal. □

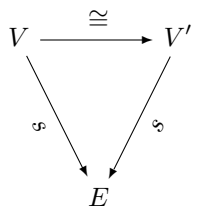
Given any subset $U \subseteq V$ in a graph with boundary Γ, the preceding lemma implies that $\Gamma - U$ is a well defined operation on graphs with boundary.

Lemma 6.20. *Let $\Gamma = (G, \partial G)$ and suppose $x \in \partial G$; then $\Gamma \cong \Gamma - x$.*

Proof. Since x is a boundary node, $\mathrm{in}(x) + \mathrm{out}(x)$ is a singleton, hence

$$(V - \{x\}) + \mathrm{in}(x) + \mathrm{out}(x) \;\cong\; (V - \{x\}) + \{*\} \;\cong\; V.$$

The edges and circles are unchanged, and



and similarly for the target map. Since the new node is a boundary node, f_V preserves the boundary as required. □

Remark. Since the isomorphism commutes with the source and target maps, and preserves the boundary, it is a morphism in the category of graphs with boundary, to be defined later.

Lemma 6.21. *If Γ is an acyclic graph with boundary, with vertex x then $\Gamma - x$ is also acyclic.*

Proof. Suppose that there is a cycle in $\Gamma - x$,

$$v_0 \xrightarrow{e_1} v_1 \xrightarrow{e_2} \cdots \xrightarrow{e_n} v_0.$$

For all i, v_i has degree at least 2, hence $v_i \notin \mathrm{in}(x)$ and $v_i \notin \mathrm{out}(x)$ (recall $\mathrm{in}(x)$ and $\mathrm{out}(x)$ are included in the boundary of $\Gamma - x$). Therefore each of the vertices on the cycle must belong to V, and hence the cycle must also be present in Γ, contradicting the assumption of acyclicity. □

Definition 6.22. Let $\Gamma = (G, \partial G)$ be a graph with boundary. Call an interior node x *peripheral* if it is adjacent to at least one other node, and either of the following hold: (i) for all other nodes y, $y \to x$ implies $y \in \partial G$; or (ii) for all other nodes y, $x \to y$ implies $y \in \partial G$.

That is, x is peripheral if either all its successors or all its predecessors are in the boundary. Note that if an acyclic graph has any interior nodes, it necessarily contains peripheral nodes.

Proposition 6.23. *Let $\Gamma = (G, \partial G)$ be an acyclic, connected graph with boundary whose interior is non-empty; then there exists a peripheral node p such that the interior of $\Gamma - p$ is acyclic and connected.*

The proof of the proposition requires some intermediate definitions.

Definition 6.24. An increasing chain $v_1 \prec v_2 \prec \cdots \prec v_n$ is called *complete* if (i) $v_i \to v_{i+1}$ for all i; and (ii) v_1 is minimal and v_n is maximal with respect to $\prec$. A sequence of vertices $\{v_i\}_{i=0}^n$ is a complete decreasing chain if $\{v_{n-i}\}_{i=0}^n$ is a complete increasing chain.

Definition 6.25. A *zig-zag* is an alternating sequence of increasing and decreasing complete chains,

$$\begin{aligned} x_1^{(1)} &\prec \cdots \prec x_{n_1}^{(1)} \\ x_1^{(2)} &\succ \cdots \succ x_{n_2}^{(2)} \\ &\vdots \\ x_1^{(k)} &\prec \cdots \prec x_{n_k}^{(k)}, \end{aligned}$$

such that, for all i, $x_{n_i}^{(i)} = x_1^{(i+1)}$, but $x_i^{(k)} \neq x^{(l)_j}$ otherwise. If Z is a zig-zag, define a ordering $\sqsubset_Z$ by

$$\begin{aligned} x_i^{(k)} &\sqsubset_Z x_j^{(l)} \text{ if } k < l \\ x_i^{(k)} &\sqsubset_Z x_j^{(k)} \text{ if } i < j. \end{aligned}$$

The length of a zig-zag is the number of distinct vertices which occur in it.

Lemma 6.26. *Let Γ be a graph with boundary with non-empty interior; then Γ contains a maximal zig-zag.*

Proof. This is an immediate consequence of the finiteness of graphs and the property of zig-zags that no vertex may appear twice. □

Proof of Proposition 6.23. Let Z be a maximal zig-zag in I_Γ, in the sense that there is no other zig-zag in I_Γ having Z as an initial segment. Let v be the maximal vertex of $\sqsubset_Z$; since v is the end-point of a complete chain it is necessarily peripheral.

Suppose $\Gamma - v$ is disconnected. All the vertices of Z must be contained in the same connected component, since they are all connected and v occurs only at the end of the sequence. Let this component be Γ'. Since Z is maximal in I_Γ all the other connected components must consist only of boundary nodes; hence the interior of $\Gamma - v$ is connected. □

In the bulk of this chapter we will be concerned with a category whose arrows are a particular class of graphs called *circuits* which are described in the next section. In order to show that certain circuits exist it is useful to consider the category whose objects are graphs with circles and boundaries. We start by rephrasing Definition 6.5 and 6.8 in more abstract terms.

Consider two finite categories:

$$\mathcal{E} = E \underset{t}{\overset{s}{\rightrightarrows}} V \qquad \text{and} \qquad \mathcal{C} = C.$$

Definition 6.27. A *graph* is functor $G : \mathcal{E} \longrightarrow \mathbf{Set}$. A *graph with circles* is a functor $H : \mathcal{E} + \mathcal{C} \longrightarrow \mathbf{Set}$.

Definition 6.28. A homomorphism between two graphs (graphs with circles) G, H is a natural transformation $f : G \Rightarrow H$ in $\mathbf{Cat}(\mathcal{E}, \mathbf{Set})$ (in $\mathbf{Cat}(\mathcal{E}+\mathcal{C}, \mathbf{Set})$).

The class of graphs forms a category, standardly denoted **Grph**. In a similar fashion, the graphs with circles form a subcategory of the functor category $\mathbf{Cat}(\mathcal{E} + \mathcal{C}, \mathbf{Set})$, which we denote $\mathcal{G}$. Let $\emptyset_C : \mathcal{C} \to \mathbf{Set}$ be the functor defined by $C \longmapsto \emptyset$. If G is a graph then $[G, \emptyset_C]$ is a graph with circles. The assignment $\cdot \mapsto [\cdot, \emptyset_C]$ extends to an embedding $K : \mathbf{Grph} \longrightarrow \mathcal{G}$. Given a graph with circles H, let UH be its restriction to $\mathcal{E}$; U extends to a forgetful functor $\mathcal{G} \to \mathbf{Grph}$ in the obvious fashion. These functors combine to give an adjunction

$$\mathbf{Grph} \underset{U}{\overset{K}{\underset{\longleftarrow}{\overset{\longrightarrow}{\bot}}}} \mathcal{G}.$$

Armed with a more abstract definition of graph we can restate Definition 6.17:

Definition 6.29. A *graph with boundary* is a pair $(G, \partial G)$ of a graph with circles G and a set ∂G such that $\partial G \subseteq GV$ and $x \in \partial G$ implies $\deg(x) = 1$.

Definition 6.30. A homomorphism of graphs with circles $f : G \to H$ defines a homomorphism of graphs with boundary $f : (G, \partial G) \to (H, \partial H)$ if $f_V(\partial G) \subseteq \partial H$.

It is immediate that if $F \xrightarrow{f} G \xrightarrow{g} H$ are morphisms which preserve boundaries then their composition must also preserve the boundary. Hence the graphs with boundary and their homomorphisms form a category which we denote $\mathcal{G}_\partial$. Note that $\mathcal{G}$ is a subcategory of $\mathcal{G}_\partial$ and the embedding functor

$$K : \mathcal{G} \hookrightarrow \mathcal{G}_\partial :: G \mapsto (G, \emptyset)$$

is left adjoint to the forgetful functor

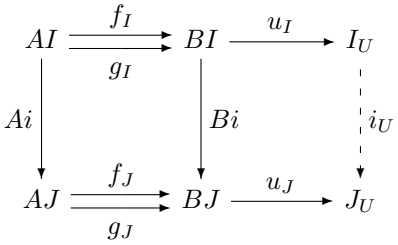
$$U : \mathcal{G}_\partial \longrightarrow \mathcal{G} :: (G, \partial G) \mapsto G.$$

Proposition 6.31. *$\mathcal{G}_\partial$ has coequalisers for all pairs of arrows.*

Proof. Suppose that we have graphs with boundary and homomorphisms

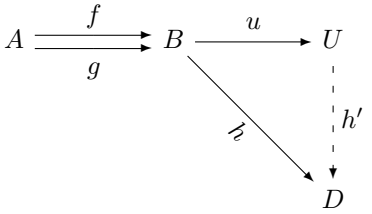
$$A \underset{g}{\overset{f}{\rightrightarrows}} B.$$

Suppose that $I \xrightarrow{i} J$ is an arrow in $\mathcal{E}$. In **Set** we have coequalisers $\langle u_I, I_U\rangle$ and $\langle u_J, J_U\rangle$ for the pairs (f_I, g_I) and (f_J, g_J). By the universal property of $\langle u_I, U_I\rangle$ the map indicated i_U below exists and is unique.



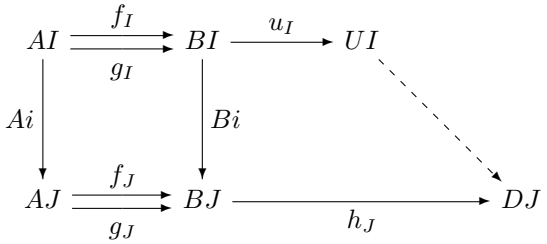
Hence define a functor $U : \mathcal{E} \longrightarrow$ **Set** by $UI = I_U$ and $Ui = i_U$ for all objects I and arrows i. The commutativity of this diagram asserts that the $\mathcal{E}$-indexed family of maps u is a natural transformation $u : B \Rightarrow U$. Let $\partial U = u_V(\partial B)$; this defines a graph $(U, \partial U)$, and a graph homomorphism $u : B \longrightarrow U$ such that $uf = ug$.

To show that $\langle u, U\rangle$ is the coequaliser for f, g it remains to prove the universal property, namely that for any map h such that $hf = hg$, there exists a unique h' such that $h = h'u$ as shown below.

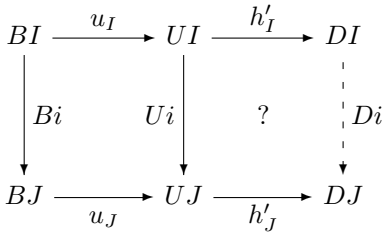


Since each component of u is a coequaliser in **Set** we have a unique map h'_I for each object I of $\mathcal{E}$ such that $h'_I u_I = h_I$. Therefore h' is the unique family of maps such that $h'u = h$; it remains to check that h' is a morphism of graphs.

By definition of U, $\partial U = u_V(\partial B)$. Since h is a homomorphism h_V sends ∂B into ∂D, and $h_V = h'_V u_V$, $h'(\partial U) \subseteq \partial D$. Again take $I \xrightarrow{i} J$ to be a morphism in $\mathcal{E}$. Consider



The right square commutes because by the naturality of f and g; therefore $h_J Bi f_I = h_J Big_I$ and so $h_J Bi$ must factor uniquely through u_I. Expanding h as $h'u$ we have



where the naturality of h makes the outer rectangle commute, and the naturality of u makes the left hand square commute. Hence $h_J Bi = Dih_I = Dih'_I u_I$ and also $h_J Bi = h'_J u_J Bi = h'_J Uiu_I$; but the factorisation through u_I is unique, hence the square marked "?" commutes, and therefore h' is natural.

Hence h' is a graph morphism, satisfying the universal property, therefore u is the coequaliser of f and g. Since f and g were arbitrary, $\mathcal{G}_\partial$ has coequalisers for any two arrows. □

Proposition 6.32. *$\mathcal{G}_\partial$ has all finite coproducts.*

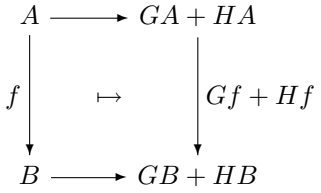
Proof. The structure lifts from coproducts in **Set**. We show that $\mathcal{G}$ has an initial object and binary coproducts: arbitrary (strict) coproducts can be constructed similarly.

Let $\underline{\emptyset} = (\emptyset, \emptyset, \emptyset, \emptyset, \emptyset)$; then $(\underline{\emptyset}, \emptyset)$ is the empty graph (with empty boundary). Since $\emptyset$ is initial in **Set**, there is a unique graph morphism

$$(\underline{\emptyset}, \emptyset) \longrightarrow (G, \partial G),$$

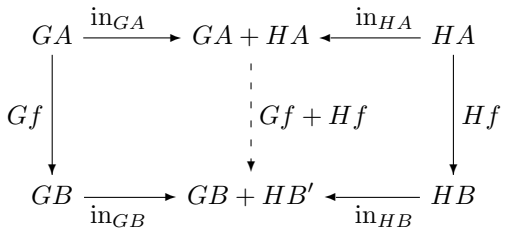
to each $(G, \partial G)$, whose components are the empty map.

Given two graphs G and H, let $G + H$ be the functor defined by



where the right hand side uses the coproduct in **Set**, and let $\partial(G + H) = \partial G + \partial H$.

By definition of the coproduct in **Set**, $Gf + Hf = [in_{GB} \circ Gf, in_{HB} \circ Hf]$ is the unique map making the diagram



commute. Hence we have natural transformations

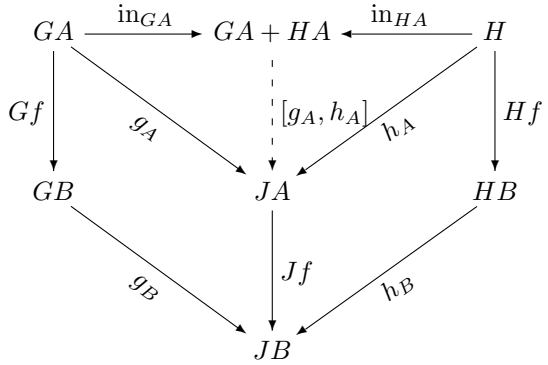
$$G \xrightarrow{\text{in}_1} G + H \xleftarrow{\text{in}_2} H$$

which, by definition, preserve the boundaries of G and H.

Suppose we have a third graph $(J, \partial J)$ and maps

$$(G, \partial G) \xrightarrow{g} (J, \partial J) \xleftarrow{h} (H, \partial H).$$

For every arrow $A \xrightarrow{f} B$ of $\mathcal{E} + \mathcal{C}$ there is a diagram in **Set**



and hence

$$\begin{aligned} Jf \circ [g_A, h_A] &= [Jf \circ g_A, Jf \circ h_A] \\ &= [g_B \circ Gf, h_B \circ Hf] \\ &= [g_B, h_B] \circ (Gf + Hf). \end{aligned}$$

This makes $[g, h]$ a natural transformation from $G + H$ to J. If $v \in \partial(G + H)$ then necessarily either $v \in \partial G$ or $v \in \partial H$; in the first case $g_V(v) \in \partial J$, and in the second $h_V(v) \in \partial J$. Therefore $[g, h]_V(v) \in \partial J$, hence $[g, h]$ is a morphism of graphs. The fact that $[g, h]$ is the unique graph morphism making the coproduct diagram commute follows from the uniqueness of each of its components in **Set**.

Therefore $\mathcal{G}$ has binary coproducts and an initial object $(\underline{\emptyset}, \emptyset)$, which suffices to provide all finite coproducts. □

Corollary 6.33. *$\mathcal{G}_\partial$ has all finite colimits.*

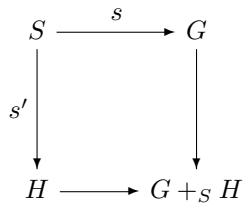
Proof. Finite coproducts and coequalisers suffice to generate all finite colimits (see [ML97]). □

Since left adjoints preserve colimits this suffices to show that both G and **Grph** are also finitely cocomplete.

While the colimit structure of $\mathcal{G}_\partial$ essentially lifts from **Set** the same is not true for limits. In particular the one point set does not give a terminal object due the need to preserve the boundary, since a graph with one vertex necessarily has an empty boundary. Nor does $\mathcal{G}_\partial$ have binary products, for essentially the same reason.

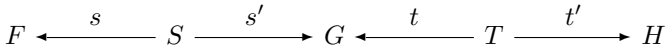
For the rest of this chapter all graphs will be directed graphs with circles and boundary. Henceforth this structure is simply called a graph.

There is an operation of "overlaying" graphs onto a common subgraph. Given a graph S with morphisms $G \xleftarrow{s} S \xrightarrow{s'} H$ then define $G +_S H$ to be the vertex of the pushout

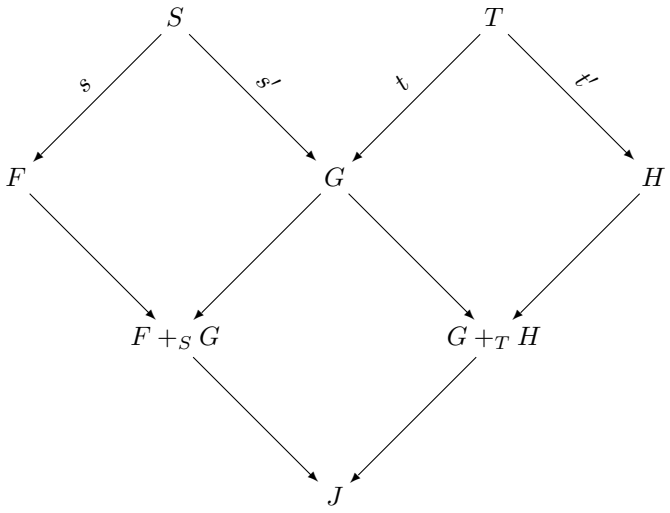


Obviously, the resulting graph depends on the morphisms s and s' but we will generally take these to be given, and use the underspecified notation $G +_S H$. The principal case of interest is when S is a common subgraph of G and H, in which case the morphisms are taken to be the canonical inclusions.

The resulting operation is clearly commutative. Given graphs and morphisms



each rectangle of the diagram



is a pushout, hence $J = (F +_S G) +_T H = F +_S (G +_T H)$.

6.2.2 Circuits

Definition 6.34. A *circuit* $\Gamma = (G, \operatorname{dom}\Gamma, \operatorname{cod}\Gamma, <_{\mathrm{in}(\cdot)}, <_{\mathrm{out}(\cdot)})$ where:

- $G = ((V, E, C, s, t), \partial G)$ is a graph;
- $\operatorname{dom}\Gamma$ and $\operatorname{cod}\Gamma$ are totally ordered sets such that $\partial G = \operatorname{dom}\Gamma + \operatorname{cod}\Gamma$;
- $<_{\mathrm{in}(\cdot)}$ is a family of maps, indexed by V such that $<_{\mathrm{in}(v)}\colon \mathrm{in}(v) \xrightarrow{\cong} \mathbb{N}_k$ where $k = |\mathrm{in}(v)|$.

- $<_{\mathrm{out}(\cdot)}$ is a family of maps, indexed by V such that $<_{\mathrm{out}(v)}\colon \mathrm{out}(v) \xrightarrow{\cong} \mathbb{N}_{k'}$ where $k' = |\mathrm{out}(v)|$.

As suggested by their name, the purpose of the two maps $<_{\mathrm{in}(\cdot)}$ and $<_{\mathrm{out}(\cdot)}$ is to impose a linear order on $\mathrm{in}(v)$ and $\mathrm{out}(v)$. Since the maps give a bijective correspondence between $\mathrm{in}(v), \mathrm{out}(v)$ and an initial segment of the naturals, the order in $\mathbb{N}$ lifts, and hence we will often simply treat these sets are ordered, and write $<$ for this ordering whenever unambiguous to do so.

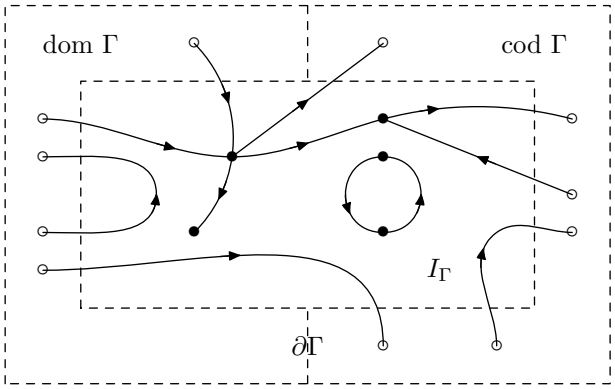


Figure 6.8: Anatomy of a circuit

A circuit is *prime* if it is connected and contains exactly one inner node, *simple* when it has no inner nodes, and *elementary* when each of its connected components is prime or simple. A circuit is *even* when its domain and codomain have the same cardinality.

Definition 6.35. A homomorphism of circuits $f : \Gamma \to \Delta$ is a graph homomorphism between the underlying graphs of Γ and Δ which preserves the additional structure. Concretely:

- if $x, y \in \mathrm{dom}\,\Gamma$ such that $x < y$ then $f_V(x), f_V(y) \in \mathrm{dom}\,\Delta$ and $f_V(x) < f_V(y)$;
- if $x, y \in \mathrm{cod}\,\Gamma$ such that $x < y$ then $f_V(x), f_V(y) \in \mathrm{cod}\,\Delta$ and $f_V(x) < f_V(y)$;
- for all vertices v in Γ, and all edges e_1, e_2 in $\mathrm{in}(v)$, if $e_1 < e_2$ in $\mathrm{in}(v)$ then $f_E(e_1) < f_E(e_2)$ in $\mathrm{in}(f_V(v))$;
- for all vertices v in Γ, and all edges e_1, e_2 in $\mathrm{out}(v)$, if $e_1 < e_2$ in $\mathrm{out}(v)$ then $f_E(e_1) < f_E(e_2)$ in $\mathrm{out}(f_V(v))$.

A map between circuits is an isomorphism if the underlying graph morphism is an isomorphism; in this case all the orders are preserved exactly.

Obviously circuits and circuit homomorphisms form a category, related by a forgetful functor to $\mathcal{G}_\partial$. There is no canonical way to derive a circuit from a graph with boundary. In this section we will define a different category, called **Circ** , which has circuits as morphisms. We now introduce some constructions on circuits which will be used in the definition of **Circ**.

Definition 6.36. Let R, S be linearly ordered sets. The standard order on their disjoint union $R + S$ is defined as

$$\begin{aligned} x < y \quad & \text{if } x \in R \text{ and } y \in S \\ x < y \quad & \text{if } x, y \in R \text{ and } x <_R y \\ x < y \quad & \text{if } x, y \in S \text{ and } x <_S y. \end{aligned}$$

For convenience we recall Definition 2.39.

Definition 6.37. A *signed set* S is a function from a carrier set $|S|$ to the set $\{+, -\}$. Given signed sets R and S, let R^* denote the signed set with the opposite signing to R; let $R \otimes S$ be the disjoint union of R and S, such that $|R \otimes S| = |R| + |S|$.

Let Γ be a circuit. Define a signing on its boundary $\partial\Gamma$ by the direction of the edges: for each $b \in \partial\Gamma$, set $b \mapsto +$ if $\mathrm{out}(v) = \emptyset$ and $b \mapsto -$ otherwise. In other words, b is positively signed if the edge incident at b goes toward b and negatively signed if the edge goes away from b. Hence $\partial\Gamma$ makes $\mathrm{dom}\,\Gamma$ and $\mathrm{cod}\,\Gamma$ isomorphic to signed finite ordinals.

Given some circuit $\Gamma = (\Gamma, \mathrm{dom}\,\Gamma, \mathrm{cod}\,\Gamma)$ define its *opposite* $\Gamma^o = (\Gamma, \mathrm{cod}\,\Gamma, \mathrm{dom}\,\Gamma)$.

Proposition 6.38. *Let* ***Circ*** *denote the set of all circuits.* ***Circ*** *forms a compact closed category.*

Proof. The objects of **Circ** are linearly ordered finite signed sets. An arrow $f : A \to B$ is a circuit with domain A^* and codomain B. Two arrows are considered equal if they are isomorphic as circuits.

Suppose we have circuits

$$A \xrightarrow{\ f\ } B \xrightarrow{\ g\ } C$$

The set B specifies a discrete graph with empty boundary, which we also denote B. We have a pair of canonical monomorphisms

$$f \hookleftarrow B \hookrightarrow g$$

in $\mathcal{G}$, which send B to the corresponding vertices of the underlying graphs (without boundary) of f and g. Every positively signed vertex B will have exactly one incoming node in f and one outgoing node in g, and vice versa for the negatively signed vertices. Hence every vertex in the image B in the pushout $f +_B g$ has in-degree 1 and out-degree 1. Hence we define the composition of circuits

$$g \circ f = ((f +_B g)_{\lhd B}, \mathrm{dom}\, f, \mathrm{cod}\, g, [<_{\mathrm{in}^f(\cdot)}, <_{\mathrm{in}^g(\cdot)}], [<_{\mathrm{out}^f(\cdot)}), <_{\mathrm{out}^g(\cdot)})]),$$

where the orders on the domain and codomain are respectively those of f and g. Given circuits

$$A \xrightarrow{\ f\ } B \xrightarrow{\ g\ } C \xrightarrow{\ h\ } D$$

Since B and C are disjoint sets of vertices in the graph of g we have

$$\begin{aligned} h \circ (g \circ f) &= (h +_C ((g +_B f)_{\lhd B}))_{\lhd C} \\ &= (h +_C g +_B f)_{\lhd B})_{\lhd C} \\ &= (h +_C g +_B f)_{\lhd C})_{\lhd B} \\ &= (((h +_C g)_{\lhd C}) +_B f)_{\lhd B} \\ &= (h \circ g) \circ f. \end{aligned}$$

Let E be the positively signed singleton. Define 1_E as the unique simple circuit with domain E^* and codomain E; let $1_{E^*} = 1_E^o$. Note that for any signed set S, we have

$$S \cong \bigotimes_i E_i,$$

where each E_i is either E or E^*, hence define $1_S = \bigotimes_i 1_{E_i}$. Clearly, $1_B f = f = f 1_A$. Hence **Circ** is a category.

Circ is strictly monoidal with tensor on signed sets and circuits defined by the coproduct of G_∂; to get a circuit we define $\mathrm{dom}(\Gamma \otimes \Delta) = \mathrm{dom}\,\Gamma \otimes \mathrm{dom}\,\Delta$ and $\mathrm{cod}(\Gamma \otimes \Delta) = \mathrm{cod}\,\Gamma \otimes \mathrm{cod}\,\Delta$, taking in both cases the standard ordering. Let $I = \emptyset$ and 1_I is the empty circuit.

Given signed sets S, T the symmetry map $s_{S,T}$ is defined as the simple circuit

$$s_{S,T} = (1_{S \otimes T}, S + T, T + S, \cdot, \cdot).$$

(Since the circuit is simple the order at each vertex is trivial.) The required symmetry condition, $s_{T,S} \circ s_{S,T} = 1_{S \otimes T}$ is immediate.

Finally, the compact closure of **Circ** is is given by taking $\eta_A : I \to A^* \otimes A$ and $\epsilon_A : A \otimes A^* \to I$ to be

$$\begin{aligned} \eta_A &= (1_A, \emptyset, A^* \otimes A) \\ \epsilon_A &= (1_A, A \otimes A^*, \emptyset). \end{aligned}$$

The required equalities follow directly from the definition of composition. □

Remark. Since every ordered finite set is order isomorphic to a finite ordinal, **Circ** is equivalent to its full subcategory determined by the signed ordinals. Hence we will assume that the objects of **Circ** are ordinals whenever convenient to do so.

Only simple circuits are required to produce the compact closed structure: therefore any subcategory of **Circ** which contains all the objects and simple circuits will also be compact closed. Indeed, the subcategory of **Circ** containing all the objects and exactly the simple circuits is the free compact closed category generated by the discrete category **1**.

Call a signed set *positive* if all its elements are mapped to $+$. Let $\mathbf{Circ}^+$ be the full subcategory of **Circ** determined by the positive objects. Call a circuit positive if it is an arrow of $\mathbf{Circ}^+$. Note that $\mathbf{Circ}^+$ is a symmetric traced monoidal category, with the canonical trace of **Circ**.

Proposition 6.39. *Every hom-set of* ***Circ*** *is isomorphic to a hom-set of* $\boldsymbol{Circ}^+$

Proof. The basic idea is to use the partial name and coname construction (Lemma 2.21) to move all positively signed nodes of the domain to the codomain, and all negatively signed nodes in the codomain to the domain.

Concretely, any object X of **Circ** is isomorphic to $X_+ \otimes X_-^*$ where X_+ and X_- are positive signed sets. Hence every hom-set $\mathbf{Circ}(X, Y)$ is isomorphic to $\mathbf{Circ}(X_+ \otimes X_-^*, Y_-^* \otimes Y_+)$. Given $f : X_+ \otimes X_-^* \to Y_-^* \otimes Y_+$, we can construct $f' : Y_- \otimes X_+ \to Y^+ \otimes X_-$ via

$$\begin{array}{ccc} Y_- \otimes X_+ & \xrightarrow{\quad f' \quad} & Y_+ \otimes X_- \\ \Big\downarrow{\scriptstyle 1 \otimes \eta_{X_-}} & & \Big\uparrow{\scriptstyle \epsilon_{Y_-} \otimes 1} \\ Y_- \otimes X_+ \otimes X_-^* \otimes X_- & \xrightarrow[1 \otimes f \otimes 1]{} & Y_- \otimes Y_-^* \otimes Y_+ \otimes X_- \end{array}$$

Since X_-^* and Y_-^* are both positive, f' is an arrow of $\mathbf{Circ}^+$. Note that

$$(1 \otimes \epsilon_{X_-}) \circ (1 \otimes f' \otimes 1) \circ (\eta_{Y_-} \otimes 1) = f,$$

hence we have an isomorphism as required. □

Corollary 6.40. *Any circuit can be canonically transformed into a positive circuit.*

Given a circuit Γ, operations $\Gamma_{\triangleleft e}$ and $\Gamma_{\triangleright v}$ lift directly from graphs to circuits. To lift $\Gamma - \{v\}$ we require an extra definition.

Definition 6.41. Let Γ be a circuit with interior vertex v. Define

$$\Gamma - \{v\} = (G - \{v\}, \operatorname{dom}\Gamma + \operatorname{in}(v), \operatorname{cod}\Gamma + \operatorname{out}(v), <_{\operatorname{in}(\cdot)}, <_{\operatorname{out}(\cdot)})$$

where the order on the domain and codomain is standard, and the order at the new vertices is trivial since they are all degree 1.

Note that by virtue of the ordering on the domain and codomain it is no longer the case that $\Gamma - V$ is well defined for sets of vertices V.

Lemma 6.42. *Let $\Gamma : A \to B$ be a circuit with two distinct vertices u, v; then*

$$(\Gamma - \{v\}) - \{u\} = (1_A \otimes \sigma) \circ ((\Gamma - \{u\}) - \{v\}) \circ (1_B \otimes \tau)$$

where σ and τ are the permutation maps

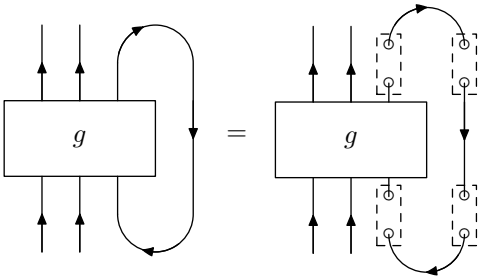
$$\begin{aligned} \sigma &: \operatorname{out}(u) \otimes \operatorname{out}(v) \longrightarrow \operatorname{out}(v) \otimes \operatorname{out}(u) \\ \tau &: \operatorname{in}(v) \otimes \operatorname{in}(u) \longrightarrow \operatorname{in}(u) \otimes \operatorname{in}(v), \end{aligned}$$

taking $\operatorname{out}(u)$ *and* $\operatorname{out}(v)$ *as positively signed, and* $\operatorname{in}(u)$ *and* $\operatorname{in}(v)$ *as negatively signed.*

Proof. Immediate from definition. □

Now consider the canonical trace on **Circ**:

$$\mathrm{Tr}_E(g) \quad = \quad (1_n \otimes \epsilon_{1^*}) \circ (g \otimes 1_{1^*}) \circ (1_m \otimes \eta_1)$$



This equality goes both ways, so given circuit f containing a cycle, we can obtain g such that $f = \mathrm{Tr}(g)$ simply by "cutting the loop". Clearly each such cutting adds one positive and one negative point to the circuit's boundary. This construction is quite general.

Given a circuit $\Gamma : A \to B$ with an interior edge $v_1 \xrightarrow{e} v_2$, we construct a new circuit Γ' as follows. Let $\mathrm{dom}\,\Gamma' = A^* \otimes E^*$ and $\mathrm{cod}\,\Gamma' = B \otimes E$; then construct the graph of Γ' as $\Gamma' = \Gamma_{\lhd e} - \{e\}$.

Lemma 6.43. *Let Γ, Γ' be circuits as described above; then $\Gamma = \mathrm{Tr}_E^{A,B}(\Gamma')$.*

Proof. By Lemma 6.10 the new vertex e has degree two, hence let b_+, b_- be the pair of corresponding points in $\partial\Gamma'$, b_- in the domain, b_+ in the codomain. To compute the trace we first form $\Gamma' \otimes 1_{E^*}$, which adds a new pair of boundary points c_+, c_-, respectively in the domain and codomain. Pre- and post-composition with η_{E^*} and ϵ_E respectively adds new edges $c_+ \to b_-$ and $b_+ \to c_-$. By the construction of Γ' there are interior points v_1, v_2 such that $v_1 \to b_+$ and $b_- \to v_2$, hence we have a composition of edges

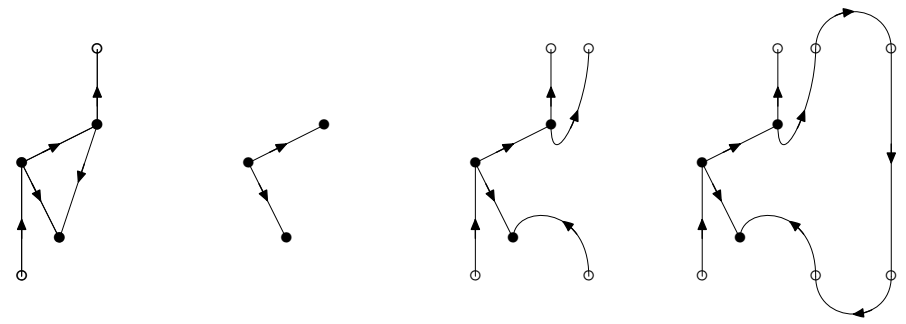
$$v_1 \longrightarrow b_+ \longrightarrow e_- \longrightarrow e_+ \longrightarrow b_- \longrightarrow v_2.$$

By the definition of composition, the intermediate vertices are deleted leaving behind an edge $v_1 \to v_2$, which was the edge of Γ originally deleted to form Γ'. □

By picking an internal edge f of Γ' and iterating this procedure we can produce another circuit $\Gamma'' = \Gamma'_{\lhd f} - \{f\}$, and so on. Call any circuit produced from Γ in this way a *skeleton* of Γ. The interior of a skeleton is a proper complete subgraph of I_Γ. If T is maximal among such subgraphs contained in a skeleton Δ then say that Δ is determined by T, or that it is a closure of T. In general T will have many closures since it depends only on the set of edges removed from Γ, while Δ depends additionally on the order of their removal.

Lemma 6.44. *Let $\Xi : A \otimes U \to B \otimes U$, $\Sigma : A \otimes U' \to B \otimes U'$ be two skeletons of a circuit $\Gamma : A \to B$, both determined by the same interior subgraph T.*

- $\mathrm{Tr}_U^{A,B}(\Xi) = \Gamma$
- $\Xi = (1_B \otimes \sigma^{-1}) \circ \Sigma \circ (1_A \otimes \sigma)$ *where σ is a permutation.*



(a) (b) (c) (d)

Figure 6.9: (a) A graph G; (b) a subgraph of G's interior; (c) a skeleton determined by the subgraph; (d) taking the trace.

Proof. Suppose $I_\Gamma - T = \{e_1, \ldots, e_n\}$ and that Ξ is produced from Γ by removing the edges in this order. Then we have a sequence of skeletons $\Gamma = \Xi_0, \Xi_1, \ldots, \Xi_n = \Xi$. Let $U = E_n \otimes \cdots \otimes E_1$ where each E_i is a signed singleton. By the vanishing rule of the trace we have

$$\mathrm{Tr}^U(\Xi) = \mathrm{Tr}^{E_1}(\cdots \mathrm{Tr}^{E_n}(\Xi) \cdots)$$

By Lemma 6.43, $\mathrm{Tr}^{E_i}(\Xi_i) = \Xi_{i-1}$, which gives the first result.

Since Ξ and Σ are generated by the same graph, the same edges $e_1, \ldots, e_n$ have been removed from Γ in a different order, say $e_{\pi(1)}, \ldots, e_{\pi(n)}$ for some permutation π on n. Hence $U' = E_{\pi(n)} \otimes \cdots \otimes E_{\pi(1)}$ and so σ is the permutation sending $i \mapsto \pi(i)$. □

Suppose $|U| = n$, and hence also $|U'| = n$. Each pair of (Γ, T) as above determines a groupoid $\mathcal{G}_{\Gamma,T}$ whose objects are the skeletons of Γ determined by T and whose arrows are elements σ of S_n such that $\sigma : \Xi \to \Sigma$ if they are related as in the lemma above. This structure is generated by the action of S_n on any chosen skeleton; the action of any $\sigma \in S_n$ on $\mathcal{G}_{\Gamma,T}$ is simply a permutation of its objects.

Corollary 6.45. *Let $\Gamma : A \to B$ be a circuit and let S, T be complete subgraphs of I_Γ. If $\Xi : A \otimes U \to B \otimes U$ and $\Sigma : A \otimes V \to B \otimes V$ are skeletons of Γ determined by S and T respectively then $\mathrm{Tr}_U(\Xi) = \mathrm{Tr}_V(\Sigma)$.*

6.3 The Free Compact Closed Category on a Polycategory

We now consider the circuits generated by a particular polycategory. In what follows, let $\mathcal{A}$ be the polycategory generated freely from some set $\mathrm{Arr}_\mathcal{A}$ of basic poly-arrows. We assume for now that there are no equations between the generators; we return to this point in sections 6.5.4 and 6.7.

Definition 6.46 (Labelling)**.** Given a polycategory $\mathcal{A}$, an *$\mathcal{A}$-labelling* for a circuit Γ is a pair of maps $\theta = (\theta_O, \theta_A)$ where

$$\theta_O : E+C \longrightarrow \mathrm{Obj}_{\mathcal{A}}$$
$$\theta_A : V \longrightarrow \mathrm{Arr}_{\mathcal{A}}$$

such that for each node f, $\mathrm{in}(f) = \langle a_1, \ldots, a_n \rangle$ and $\mathrm{out}(f) = \langle b_1, \ldots, b_m \rangle$ imply

$$\mathrm{dom}(\theta f) = \theta a_1, \ldots, \theta a_n$$
$$\mathrm{cod}(\theta f) = \theta b_1, \ldots, \theta b_m,$$

and subject to the further restriction that $\theta_A(v) = 1_A$ if and only if $v \in \partial\Gamma$. Call a circuit Γ *$\mathcal{A}$-labellable* if there exists an $\mathcal{A}$-labelling for it; if θ is a labelling for Γ, then the pair (Γ, θ) is an *$\mathcal{A}$-labelled circuit*.

Remark. Note that the boundary nodes are not treated on the same basis as the interior nodes. We will often assume the boundary is labelled with objects of $\mathcal{A}$ rather than the corresponding identity maps, and treat the boundary as an $\mathcal{A}$-labelled signed set.

We denote the class of $\mathcal{A}$-labelled circuits $\mathbf{Circ}(\mathcal{A})$.

Proposition 6.47. *The $\mathcal{A}$-labelled circuits form a compact closed category.*

Proof. The structure lifts from the underlying category **Circ**, with the restriction that arrows $(f, \theta), (g, \kappa)$ are only composable when $\theta(\mathrm{cod}\, f) = \kappa(\mathrm{dom}\, g)$. We note that every simple circuit is $\mathcal{A}$-labellable hence the resulting subcategory is compact closed. □

We now define the canonical embedding $\Psi : \mathcal{A} \hookrightarrow \mathbf{Circ}(\mathcal{A})$. Let $\Psi_g : \mathrm{Arr}_{\mathcal{A}} \to \mathbf{Circ}(\mathcal{A})$ be defined to send $f : \otimes_{i=1}^{n} A_i \to \otimes_{j=1}^{m} B_j$ to the prime circuit with boundary $|n|^* \otimes |m|$, and its unique inner node labelled by f. Ψ_g extends in the evident way to a symmetric monoidal functor $\Psi : \mathcal{A} \to \mathbf{Circ}(\mathcal{A})$, which is well defined since, by hypothesis, there are no equations between the generators of $\mathcal{A}$.

Note that this embedding sends the objects of $\mathcal{A}$ to positive signed sets. Let $\mathbf{Circ}(\mathcal{A})^+$ be the full subcategory of $\mathbf{Circ}(\mathcal{A})$ determined by the objects of $\mathcal{A}$, or equivalently the subcategory of $\mathbf{Circ}(\mathcal{A})$ determined by $\mathbf{Circ}^+$. Proposition 6.39 lifts immediately to give the following.

Lemma 6.48. *$\boldsymbol{Circ}(\mathcal{A})^+$ is traced by the canonical trace of $\boldsymbol{Circ}(\mathcal{A})$, and every hom-set of $\boldsymbol{Circ}(\mathcal{A})$ is isomorphic to a hom-set of $\boldsymbol{Circ}(\mathcal{A})^+$.*

Lemma 6.49. *Let (Γ, θ) be a circuit of $\boldsymbol{Circ}(\mathcal{A})^+$. Γ is acyclic and connected if and only if $(\Gamma, \theta) = \Psi f$ for some unique f in $\mathcal{A}$.*

Proof. Suppose f is an arrow of $\mathcal{A}$; then f is a formal composition of generators $\mathrm{Arr}_{\mathcal{A}}$, possibly with domains and codomains permuted. By the definition of Ψ each generator g is mapped to a prime circuit in $\mathbf{Circ}(\mathcal{A})^+$. The functor Ψ commutes with composition, and composition in $\mathbf{Circ}(\mathcal{A})^+$ preserves acyclicity, since a composition of positive circuits cannot form a loop. Composition of connected circuits Γ, Γ' also preserves connectedness, provided it is not along I; but in that case the corresponding poly-arrows would not be composable. Hence Ψf is acyclic and connected.

Conversely, suppose Γ is both connected and acyclic. We use induction on the number of interior nodes of Γ. If Γ has no interior nodes then it consists of a single wire, labelled by some object $\theta_O e$; in this case $(\Gamma, \theta) = \Psi 1_{\theta_O e}$. Otherwise Γ has $n \geq 1$ interior nodes. Since it is acyclic and connected, $\Gamma = \Xi_1 \circ \Gamma_1$ or $\Gamma_2 = \Gamma_2 \circ \Xi_2$ where Ξ_1, Ξ_2 are prime circuits, and Γ_1, Γ_2 have $n - 1$ interior nodes. (This is a consequence of Lemma 6.23.)

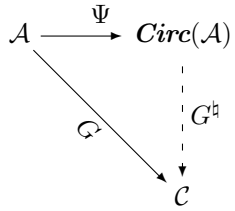
If Ξ is a prime circuit, then the labelling on its only node, v, gives $\Psi\theta v = (\Xi, \theta)$, up to a permutation on its domain and codomain. Hence if $\Xi_1 = \Xi_2$ are both prime then $\theta_1 = \theta_2$, hence $\Xi_1 = \Xi_2 = \Psi f$. By induction hypothesis there is a unique poly-arrow g_1 such that $\Gamma_1 = \Psi g_1$; hence

$$\Gamma = \Xi \circ \Gamma_1 = \Psi f \circ \Psi g_1 = \Psi(f \circ g_1),$$

and similarly if $\Gamma = \Gamma_2 \circ \Xi_2$. Note that the choice of Ξ will not usually be unique, but whenever there are conflicting choices, they will commute due to Equations (6.7) and (6.8). □

Corollary 6.50. *The functor $\Psi : \mathcal{A} \to \boldsymbol{Circ}(\mathcal{A})$ is faithful.*

Theorem 6.51. *Given a compact closed category $\mathcal{C}$, a symmetric polycategory with multicut $\mathcal{A}$ freely constructed from generators $\mathrm{Obj}_{\mathcal{A}}$ and $\mathrm{Arr}_{\mathcal{A}}$, and a symmetric monoidal functor $G : \mathcal{A} \to \mathcal{C}$, there is a unique (up to isomorphism) compact closed functor $G^\natural$ such that $G \cong G^\natural \Psi$.*



Proof. The value of $G^\natural$ is on the generators is fixed by G; thereafter its other values are fixed by the need to preserve the compact closed structure of $\mathbf{Circ}(\mathcal{A})$.

If $A \in \mathrm{Obj}_{\mathcal{A}}$ then let $G^\natural \Psi A = GA$. By definition $G^\natural((\Psi A)^*) = (GA)^*$ and $G^\natural \eta_{\Psi A} = \eta_{GA}$ and $G^\natural \epsilon_{\Psi A} = \epsilon_{GA}$. Define $G^\natural(A \otimes B) = G^\natural A \otimes G^\natural B$; since we have $(A \otimes B)^* = A^* \otimes B^*$ in $\mathbf{Circ}(\mathcal{A})$, this determines the value of $G^\natural$ on all objects.

Since compact closed functors preserve units and counits, the value of $G^\natural$ on arrows is determined by its values on $\mathbf{Circ}(\mathcal{A})^+$. Let $g = (\Gamma, \theta) \in \mathbf{Circ}(\mathcal{A})^+(A, B)$. We use a recursion on the number of cycles in the graph of g to define $G^\natural g$.

If Γ is acyclic, then $g = \Psi f$ for some $f \in \mathcal{A}(A, B)$, hence let $G^\natural g$ be given by

$$G^\natural A \xrightarrow{\cong} GA \xrightarrow{Gf} GB \xrightarrow{\cong} G^\natural B$$

noting that $G^\natural(A \otimes B) = GA \otimes GB$ on the objects of $\mathbf{Circ}(\mathcal{A})^+$.

Suppose, on the other hand, that Γ has a cycle. We can obtain a new arrow by picking an edge e which lies on the cycle and is labelled by C, then let $g' = g_{\triangleright e} - \{e\} : A \otimes C \to B \otimes C$. By Lemma 6.43 $g = \mathrm{Tr}^C_{A,B}(g')$ and hence we must have $G^\natural g = \mathrm{Tr}^{G^\natural C}_{G^\natural A, G^\natural B}(G^\natural g')$. Dinaturality of trace implies that the choice of which edge to cut is unimportant; superposing and yanking axioms

together imply the order the cycles are removed in is irrelevant. Hence $G^\natural g$ is well defined.

Clearly $G \cong G^\natural\Psi$. Suppose that $H : \mathbf{Circ}(\mathcal{A}) \to \mathcal{C}$ is a compact closed functor such that $G \cong H\Psi$; then let τ be the natural isomorphism $G^\natural\Psi \xrightarrow{\cong} H\Psi$. Since τ is invertible, let $\rho_A = (\tau_A^{-1})^*$. For every object A of $\mathbf{Circ}(\mathcal{A})$, the symmetry of the tensor gives a natural isomorphism $A \longrightarrow (\Psi A_+) \otimes (\Psi A_-)^*$, hence we have

$$G^\natural A \xrightarrow{\cong} (G^\natural\Psi A_+) \otimes (G^\natural\Psi A_-)^* \xrightarrow{\tau_{A_+} \otimes \rho_{A_-}} (H\Psi A_+) \otimes (H\Psi A_-)^* \xrightarrow{\cong} HA.$$

Therefore $H \cong G^\natural$. □

Corollary 6.52. *The inclusion of categories* ***Com*** $\hookrightarrow$ ***PolyCat*** *has a left adjoint, whose unit has component at* $\mathcal{A}$ *given by* $\Psi : \mathcal{A} \to \boldsymbol{Circ}(\mathcal{A})$.

6.4 Scalars

As we saw in Chapter 4, the structure of a compact closed category cannot be accurately represented without taking account of the scalars. This is particularly important for a logical presentation since at the level of types an arrow cannot be distinguished from its scalar multiples. In this section we characterise the scalars of **Circ** and lift those structures to $\mathbf{Circ}(\mathcal{A})$. Since the scalars form a commutative monoid with respect to the tensor, we describe only those circuits with a single connected component.

The unit object I of **Circ** is the empty signed set, hence any circuit with an empty boundary is a scalar. There are three situations where this may occur.

The simplest graph with empty boundary is a circle. It is simply a closed loop without any vertex. We set aside this special case for now, and consider only scalars which have vertices.

Suppose that Γ is a circuit of **Circ** with at least one vertex, and further assume that it is acyclic. We can view such a graph as a partial order on its vertices, with the order relation given by the connectivity of the graph. Γ is a scalar if and only if its maximal vertices all have out-degree zero, and its minimal vertices all have in-degree zero. If Γ is $\mathcal{A}$-labellable then, by Lemma 6.49, each labelling of Γ determines a unique polyarrow $f : - \to -$ in $\mathcal{A}$. In other words, the first class of scalars in $\mathbf{Circ}(\mathcal{A})$ are the scalars lifted from $\mathcal{A}$. (Of course, the scalars of $\mathbf{Circ}(\mathcal{A})$ are simply the set of polyarrows with empty domain and codomain – there is no associated algebraic structure.)

Now suppose, on the other hand, that Γ is cyclic. The following is immediate.

Lemma 6.53. *Every skeleton of a scalar is positive and even.*

Therefore, every skeleton of Γ determines an element Ξ of $\mathbf{Circ}^+$ such that $\Gamma = \mathrm{Tr}(\Xi)$. Suppose that Γ is $\mathcal{A}$-labellable by some labelling θ; then θ also gives a labelling for Ξ, which we write abusively also as θ. If Ξ is acyclic and connected then (Ξ, θ) determines a unique polyarrow f such that $(\Gamma, \theta) = \mathrm{Tr}(\Psi f)$. This will allow us to pin down the remaining scalars in the style of Section 2.8. Call a polyarrow an *endomorphism* if it has the same sequence of objects for its domain and codomain; say that two endomorphisms $f : \Gamma \to \Gamma$ and $g : \Delta \to \Delta$ are *trace equivalent* if $\mathrm{Tr}_\Gamma(\Psi f) = \mathrm{Tr}_\Delta(\Psi g)$. The scalars of $\mathbf{Circ}(\mathcal{A})$ generated

by cyclic circuits are isomorphic to the trace equivalence classes of $\mathcal{A}$. If $\mathcal{A}$ is just a category then this definition reduces to that of Section 2.8.

The vanishing axiom of the trace gives $\mathrm{Tr}_I(f) = f$, hence the acyclic scalars discussed above are included in this definition. Note that $\mathrm{Tr}_A(1_A)$ has the graph structure of a circle, so circuits with no vertices fall under this case too, and these are simply the dimension maps of the objects of $\mathcal{A}$. Hence we can sum up the preceding discussion in the following:

Proposition 6.54. *The scalars $I \to I$ of $\boldsymbol{Circ}(\mathcal{A})$ are in bijective correspondence with the trace equivalence classes of $\mathcal{A}$.*

Given a polyarrow f, the circuit for $\mathrm{Tr}(f)$ can easily be constructed, and from this circuit all the other elements of f's trace equivalence class can be computed.

We consider one special case before moving on. Suppose $\Gamma : A \to A$ is prime, and denote its single vertex x. Call a prime circuit *regular* if it is positive and acyclic, and the ordering on its domain and codomain agrees with $\mathrm{in}(x)$ and $\mathrm{out}(x)$ respectively. Such a circuit is completely fixed by the object A. If another prime $\Xi : A \to A$ is also positive, even and acyclic then there are permutations σ, σ' such that $\Xi = \sigma \circ \Gamma \circ \sigma'$, and further each pair σ, σ' determines a valid circuit in this way. Suppose that s is a prime scalar with $\Xi : A \to A$, a acyclic prime, as skeleton. By Lemma (6.44) we have $s = \mathrm{Tr}_A(\Xi)$ from whence $s = \mathrm{Tr}_A(\sigma \circ \Gamma \circ \sigma') = \mathrm{Tr}_A(\tau \circ \Gamma)$ where $\tau = \sigma'\sigma$. This leads directly to the following lemma.

Lemma 6.55. *The set of prime scalars in $\boldsymbol{Circ}$ is isomorphic to*

$$\bigcup_{n \in \mathbb{N}} \{\sigma | \sigma \in S_n\}.$$

To sum up the lemma in a slogan, the structure of a prime scalar is nothing more than a description of how the outputs" of a circuit are fed back to its "inputs".

Remark. This general statement holds in the slightly more complicated situation of non-prime scalars. Let Γ be a scalar which is not prime; let its vertices be labelled $f_1, \ldots, f_n$. By Lemma (6.44) it is equal to the trace over any of its skeletons. Let Ξ be a skeleton determined by the totally disconnected subgraph; then $\Xi = \sigma \circ (\bigotimes_i f_i)\sigma'$, where f_i denotes the regular prime determined by the vertex of the same name. This result is essentially the same as Theorem 1 of [KSW97].

6.5 Homotopy

Up to this point we have considered circuits based on a polycategory $\mathcal{A}$, whose arrows are presented as a set of generators. Under this regime, only the generators may occur as vertex labels in $\mathbf{Circ}(\mathcal{A})$. This restriction may be relaxed by considering circuits under *homotopy equivalence*. Informally, two topological spaces are homotopy equivalent if they are images of the same space under a continuous deformation. In this section we use a specialised definition for graphs; for a full treatment see e.g. [Hat02].

Throughout this section the only circuits of concern are those with at least one interior vertex. To avoid ceaseless repetition, this qualification will be omitted from results and their proofs. In all cases, if the circuit has empty interior then the result is trivial.

We proceed as follows: the class of $\mathcal{A}$-labelled circuits is expanded by adjoining formal generators for every polyarrow of $\mathcal{A}$; the notion of *contraction* is defined for labelled circuits, and hence also a notion of homotopy equivalence; we show that the quotient of the expanded class of circuits by homotopy equivalence is equivalent to the original category.

6.5.1 Extended Labellings

Definition 6.56. An *extended $\mathcal{A}$-labelling* on a circuit Γ is defined as in Definition 6.46, with the relaxation that θ_A may range over all polyarrows of $\mathcal{A}$, i.e. the restriction to generators is dropped. The resulting class of circuits is denoted $\mathbf{Circ}_{\mathrm{x}}(\mathcal{A})$.

The polyarrows of $\mathcal{A}$ are generated freely from some set $\mathrm{Arr}_{\mathcal{A}}$, so writing $F\mathrm{Arr}_{\mathcal{A}}$ for the set of polyarrows, we have $\mathcal{A} \cong (O, F\mathrm{Arr}_{\mathcal{A}})$ where O is the set of objects; then $\mathbf{Circ}(\mathcal{A}) = \mathbf{Circ}(O, F\mathrm{Arr}_{\mathcal{A}})$. Analogously $\mathbf{Circ}_{\mathrm{x}}(\mathcal{A}) = \mathbf{Circ}(O, FF\mathrm{Arr}_{\mathcal{A}})$. Hence $\mathbf{Circ}_{\mathrm{x}}(\mathcal{A})$ is simply the original **Circ** construction over an enlarged polycategory, therefore all our earlier results for $\mathbf{Circ}(\mathcal{A})$ apply to $\mathbf{Circ}_{\mathrm{x}}(\mathcal{A})$ as well.

The categories $\mathbf{Circ}(\mathcal{A})$ and $\mathbf{Circ}_{\mathrm{x}}(\mathcal{A})$ have the same objects and since $F\mathrm{Arr}_{\mathcal{A}} \subset FF\mathrm{Arr}_{\mathcal{A}}$, there is a faithful embedding

$$\mathcal{A} \xrightarrow{\Psi} \mathbf{Circ}(\mathcal{A}) \xhookrightarrow{\Psi_x} \mathbf{Circ}_{\mathrm{x}}(\mathcal{A})$$

which preserves the compact closed structure of $\mathbf{Circ}(\mathcal{A})$. The embedding is the identity on objects, so it also gives an inclusion on the traced subcategories determined by the positive objects:

$$\mathbf{Circ}^{+}(\mathcal{A}) \xhookrightarrow{\Psi_x} \mathbf{Circ}^{+}_{\mathrm{x}}(\mathcal{A}).$$

As in $\mathbf{Circ}(\mathcal{A})$, the compact closed structure of $\mathbf{Circ}_{\mathrm{x}}(\mathcal{A})$ is determined by this positive subcategory so most of this section will be devoted to $\mathbf{Circ}^{+}_{\mathrm{x}}(\mathcal{A})$ rather than the whole category.

6.5.2 Homotopy Equivalence

Definition 6.57. A graph G is contractible to G' if G' is obtained from G by identifying two adjacent vertices along an edge connecting them. Homotopy equivalence for graphs is the reflexive, symmetric, transitive closure of this relation.

From this definition, any acyclic connected graph (without parallel edges) is homotopy equivalent to a single vertex; this leads naturally to the following.

Proposition 6.58. *The interior I_G of a connected graph with boundary $(G, \partial G)$ is homotopy equivalent to single vertex with some number of self loops.*

Proof. Let T be any minimum spanning tree of I_G; then T is homotopy equivalent to a point. The quotient map $I_G \to I_G/T$ is a homotopy equivalence (see e.g. [Hat02]) and I_G/T is a graph with only one vertex, hence all its edges are self loops. □

In order to apply this notion to labelled circuits, we must extend the definition of contraction.

Definition 6.59. Let v, u be distinct interior vertices of some circuit Γ, with F a set of edges from v to u; define Γ/F, the *contractum along* F, as the circuit whose underlying graph is

$$G' = ((V - \{v, u\}) + \{\{v, u\}\},\ E - F,\ C, s', t')$$

$$s' : e \mapsto \begin{cases} \{v, u\} \text{ if } s(e) = v \text{ or } u \\ s(e) \text{ otherwise} \end{cases}$$
$$t' : e \mapsto \begin{cases} \{v, u\} \text{ if } t(e) = v \text{ or } u \\ t(e) \text{ otherwise} \end{cases}$$

the ordered set $\mathrm{out}(\{v, u\})$ and $\mathrm{in}(\{v, u\})$ are defined as

$$\begin{aligned} \mathrm{out}(\{v, u\}) &= O_1 + \mathrm{out}(u) + O_2 \\ \mathrm{in}(\{v, u\}) &= I_1 + \mathrm{in}(v) + I_2 \end{aligned}$$

where I_1 and I_2 form a partition of $\mathrm{in}(u) - F$ such that I_1 contains the edges which are ordered before all of F, and O_1, O_2 form a similarl partition of $\mathrm{out}(v)$.

Definition 6.60. With Γ, v, u, F as above, a *contraction* is a rewrite from Γ to Γ/F; if F contains all the edges from v to u it is called *maximal*.

Given vertices v, u as above, let τ and σ be permutations on $|\mathrm{out}(v)|$ and $|\mathrm{in}(u)|$ such that:

$$\begin{aligned} \tau &: \mathrm{out}(v) \mapsto O_1, F, O_2 \\ \sigma &: \mathrm{in}(u) \mapsto I_1, F, I_2 \end{aligned}$$

Lemma 6.61. *Let $\Gamma, v, u, F, \tau, \sigma$ be as above, and let Γ' be the circuit obtained by contracting along F; if θ is an $\mathcal{A}$-labelling for Γ there exists an $\mathcal{A}$-labelling of Γ'.*

Proof. We construct a canonical labelling θ' for Γ'. The edges of Γ' are a subset of those of Γ, so let $\theta'_O = \theta_O$. Suppose $\theta_A(v) = f$ and $\theta_A(u) = g$; define θ'_A by setting $\theta'_A(x) = \theta_A(x)$ for $x \notin \{v, u\}$ and $\theta'_A(w) = g_\sigma \circ f^\tau$. □

In a circuit a vertex v is *k-contractible* to v' if there exists a set of edges of cardinality k from v to v', for $k > 0$. A circuit Γ is contractible to Γ' if Γ' can be obtained by a k-contraction of Γ for any k. A labelled circuit (Γ, θ) is contractible to (Γ', θ') if Γ is contractible to Γ' and θ' is the canonical labelling of Lemma 6.61.

Since we work with symmetric polycategories the order of the edges incident at a given vertex is essentially arbitrary. Another class of rewrites internalises this symmetry.

Definition 6.62. Let v be a vertex of a circuit Γ; say that Γ *shuffles* to Γ' if Γ' is obtained from Γ by permuting the order of $\mathrm{in}(v)$ and $\mathrm{out}(v)$.

Lemma 6.63. *Let (Γ, θ) be an $\mathcal{A}$-labelled circuit such that Γ rewrites to Γ' by a shuffle at some vertex v; then Γ' is $\mathcal{A}$-labellable.*

Proof. We define a canonical labelling θ'. Suppose that

$$\begin{aligned}\mathrm{in}(v') =& \sigma(\mathrm{in}(v)),\\ \mathrm{out}(v') =& \tau(\mathrm{out}(v)),\end{aligned}$$

for some permutations σ, τ. Define the canonical labelling for θ' by $\theta'(x) = \theta(x)$ for $x \neq v$ and $\theta'(v) = (\theta(v))^{\tau}_{\sigma}$. □

Define *homotopy equivalence* of $\mathcal{A}$-labelled circuits to be the transitive, reflexive, symmetric closure of contractibility and shuffle; write $(\Gamma, \theta) \simeq (\Delta, \pi)$ for this relation. Since neither contraction nor shuffle has an effect on the boundary of a circuit, if two circuits are homotopy equivalent then they have the same type.

Lemma 6.64. *Let a circuit Γ rewrite to Γ' by some sequence of contractions or shuffles; if Γ contains a cycle then so does Γ'.*

Proof. Shuffle has no effect on the underlying graph of the circuit, so it is discounted immediately. Suppose that some sequence of vertices $v_1, \ldots, v_n, v_1$ form a cycle in Γ. If an edge $v_i \to v_{i+1}$ connecting two vertices is removed by contraction, then v_i and v_{i+1} are identified, hence the cycle is preserved. In the extreme case, all the vertices are identified leaving a loop at the sole remaining vertex. □

Lemma 6.65. *If a circuit of $\boldsymbol{Circ}_x(\mathcal{A})$ is connected, then it is homotopy equivalent to a prime circuit. If the original circuit is acyclic it is homotopy equivalent to a regular prime.*

Proof. If the graph is connected, then all the vertices are at least 1-contractible; the result is immediate from Proposition 6.58. If the original circuit is acyclic, then suppose there are k edges from vertex x to y. Contracting along all k edges will not introduce a cycle, hence by performing such maximal contractions between every adjacent pair of vertices, the resulting prime $(P, \cdot \mapsto f)$ is also acyclic. Any acyclic prime can rewrite to a regular prime by a shuffle, so we may take P to be regular. □

Lemma 6.66. *Let (Γ, θ) and (Γ', θ') be acyclic, connected circuits of $\boldsymbol{Circ}^{+}_x(\mathcal{A})$ such that (Γ, θ) rewrites to (Γ', θ') by a sequence of contractions; then (Γ', θ') depends only on the sets of vertices identified.*

Proof. Suppose there is a sequence of contractions:

$$(\Gamma, \theta) = (\Gamma_0, \theta_0) \xrightarrow{c_1} (\Gamma_1, \theta_1) \xrightarrow{c_2} \cdots \xrightarrow{c_n} (\Gamma_n, \theta_n) = (\Gamma', \theta').$$

Each contraction unifies two vertices, so we may view the vertices of each (Γ_i, θ_i) as sets of vertices of (Γ, θ). Each contraction can be seen as a union of these sets. We show that the order of any pair of contractions may be exchanged

without changing (Γ', θ'). Evidently such reordering has no effect on the sets of vertices at the final stage and generates all possible contraction sequences.

We use induction on the length of the sequence of contractions.

If $n = 0$ then $(\Gamma, \theta) = (\Gamma', \theta')$ and the result is trivial. If $n = 1$, there is only one possible choice of contraction since, if v and u are the vertices to be contracted, either $v \to u$ or $u \to v$ but not both, because (Γ, θ) is acyclic. Further, the contraction must be maximal since any non-maximal contraction would introduce a cycle, contradicting the acyclicity of (Γ', θ').

Otherwise, consider an adjacent pair of rewrites in the sequence,

$$(\Gamma_{i-1}, \theta_{i-1}) \xrightarrow{c_i} (\Gamma_i, \theta_i) \xrightarrow{c_{i+1}} (\Gamma_{i+1}, \theta_{i+1})$$

c_i, c_{i+1}. If they act on disjoint sets of vertices, then they may be exchanged without affecting $(\Gamma_{i+1}, \theta_{i+1})$. Otherwise, there are four cases, depending on the topology of the three vertices in Γ_{i-1} which will be identified.

In the diagrams below, suppose c_i contracts along the edges labelled 1, and c_{i+1} along those labelled 2. (Only one edge is shown between the vertices, but the contractions are necessarily maximal, so no information is lost.)

Case 1

$$f \xrightarrow{1} g \xrightarrow{2} h$$

Contraction along 1 produces a new vertex labelled by $g_{\sigma_1} \circ f^{\tau_1}$, for some permutations σ_1, τ_1 making f, g composable. Subsequent contraction along along 2 gives a vertex labelled by $h_{\sigma_2} \circ (g_{\sigma_1} \circ f^{\tau_1})^{\tau_2}$. Since f and h are not adjacent the permutation τ_2 acts only on the codomain of g, hence

$$h_{\sigma_2} \circ (g_{\sigma_1} \circ f^{\tau_1})^{\tau_2} = h_{\sigma_2} \circ (g_{\sigma_1}^{\tau_2} \circ f^{\tau_1})$$

Similarly contracting 2 then 1 produces a vertex labelled by $(h_{\sigma_2} \circ g^{\tau_2})_{\sigma_1} \circ f^{\tau_1}$; here σ_1 has no action on the domain of h, hence

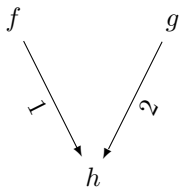
$$(h_{\sigma_2} \circ g^{\tau_2})_{\sigma_1} \circ f^{\tau_1} = (h_{\sigma_2} \circ g_{\sigma_1}^{\tau_2}) \circ f^{\tau_1},$$

which gives

$$(h_{\sigma_2} \circ g_{\sigma_1}^{\tau_2}) \circ f^{\tau_1} = h_{\sigma_2} \circ (g_{\sigma_1}^{\tau_2} \circ f^{\tau_1})$$

by associativity.

Case 2



Contraction along 1 then 2 produces a single vertex labelled by $(h_{\sigma_1} \circ f^{\tau_1})_{\sigma_2} \circ g^{\tau_2}$. Since g and f are not adjacent, σ_2 does not act on the domain of f, hence $(h_{\sigma_1} \circ f^{\tau_1})_{\sigma_2} \circ g^{\tau_2} = (h_{\sigma_1 \sigma_2} \circ f^{\tau_1}) \circ g^{\tau_2}$ from which the commutativity condition (Equations (6.7)) gives

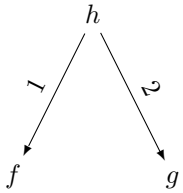
$$(h_{\sigma_1\sigma_2} \circ f^{\tau_1}) \circ g^{\tau_2} = ((h_{\sigma_1\sigma_2} \circ g^{\tau_2}) \circ f^{\tau_1})_\rho$$

where ρ is a permutation, and since σ_1 does not affect the domain of g, we have

$$((h_{\sigma_1\sigma_2} \circ g^{\tau_2}) \circ f^{\tau_1})_\rho = ((h_{\sigma_2} \circ g^{\tau_2})_{\sigma_1} \circ f^{\tau_1})_\rho$$

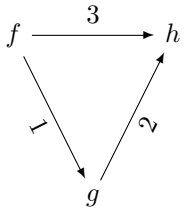
which is the label for the vertex produced by contracting along 2 then 1.

Case 3



This case is essentially the same as the previous one.

Case 4



Note that contraction along 3 is not permitted since it would introduce a cycle; the only possibilities are 1 then 2+3, or 2 then 1+3. Let the types of f, g, h be

$$\begin{aligned} f &: \Gamma \longrightarrow \Delta_1, \Phi, \Delta_2, \Psi, \Delta_3 \\ g &: \Gamma_1\Phi, \Gamma_2 \longrightarrow \Delta_4, \Xi, \Delta_5 \\ h &: \Gamma_3, \Xi, \Gamma_4, \Psi, \Gamma_5 \longrightarrow \Delta \end{aligned}$$

where Φ is the type of the edges labelled by 1, Ξ is the type of 2 and Ψ is the type of 3. We assume, without loss of generality, that no permutation is needed to make these bundles of edges composable. Suppose that edge 1 is contracted first: the resulting vertex is labelled by

$$g \circ f : \Gamma_1, \Gamma, \Gamma_2 \longrightarrow \Delta_1, \Delta_4, \Xi, \Delta_5, \Delta_2, \Psi, \Delta_3.$$

The permutations required to make $g \circ f$ and h composable are

$$\begin{aligned} \tau_1 : \Delta_1, \Delta_4, \Xi, \Delta_5, \Delta_2, \Psi, \Delta_3 &\longmapsto \Delta_1, \Delta_4, \Xi, \Psi, \Delta_5, \Delta_2, \Delta_3 \\ \sigma_1 : \Gamma_3, \Xi, \Psi, \Gamma_4, \Gamma_5 &\longmapsto \Gamma_3, \Xi, \Gamma_4, \Psi, \Gamma_5 \end{aligned}$$

hence the result of contracting along 1 then 2+3 is

$$h_{\sigma_1} \circ (g \circ f)^{\tau_1} : \Gamma_3, \Gamma_1, \Gamma, \Gamma_2, \Gamma_4, \Gamma_5 \longrightarrow \Delta_1, \Delta_4, \Delta, \Delta_5, \Delta_2, \Delta_3$$

Suppose instead that edges 2 are contracted first. The label of the resulting vertex is

$$(h \circ g) : \Gamma_3, \Gamma_1\Phi, \Gamma_2, \Gamma_4, \Psi, \Gamma_5 \longrightarrow \Delta_4, \Delta, \Delta_5,$$

and the required permutations are

$$\tau_2 : \Delta_1, \Phi, \Delta_2, \Psi, \Delta_3 \longmapsto \Delta_1, \Phi, \Psi, \Delta_2, \Delta_3$$
$$\sigma_2 : \Gamma_3, \Gamma_1\Phi, \Psi, \Gamma_2, \Gamma_4, \Gamma_5 \longmapsto \Gamma_3, \Gamma_1\Phi, \Gamma_2, \Gamma_4, \Psi, \Gamma_5$$

hence the vertex obtained by contracting 1+3 is labelled with the map

$$(h \circ g)_{\sigma_2} \circ f^{\tau_2} : \Gamma_1, \Gamma, \Gamma_2, \Gamma_4, \Gamma_5 \longrightarrow \Delta_3, \Delta_1, \Delta_4, \Delta, \Delta_5, \Delta_2, \Delta_3.$$

Notice that τ_1 factorises into $\tau_1 = \tau_1''\tau_1'$ with

$$\tau_1' : \Delta_2, \Psi \longmapsto \Psi, \Delta_2,$$
$$\tau_1'' : \Delta_5\Psi \longmapsto \Psi, \Delta_5$$

where τ_1' acts only on the domain of f; note that $\tau_1' = \tau_2$. Similarly $\sigma_2 = \sigma_2'\sigma_2''$ with

$$\sigma_2' : \Psi, \Gamma_4 \longmapsto \Gamma_4, \Psi$$
$$\sigma_2'' : \Psi, \Gamma_2 \longmapsto \Gamma_2, \Psi,$$

and $\sigma_1 = \sigma_2'$. Hence

$$(h \circ g)_{\sigma_2} \circ f^{\tau_2} = (h_{\sigma_2'} \circ g)_{\sigma_2''} \circ f^{\tau_2}$$

and

$$h_{\sigma_1} \circ (g \circ f)^{\tau_1} = h_{\sigma_2'} \circ (g \circ f^{\tau_2})^{\tau_1''}$$

and these are equal by the generalised associativity axiom.

Hence the resulting circuit is independent of the order of contractions. □

Corollary 6.67. *Let $(\Gamma.\theta)$ be an $\mathcal{A}$-labelled circuit, with H an acyclic subgraph of its interior; then operation Γ/H is well-defined.*

Proof. We wish to contract the entire subgraph H to one vertex. To do so, perform maximal (with respect to the edges of H) contractions between each pair of vertices of H; by the preceding lemma, the result is independent of the order of the contractions. □

Lemma 6.68. *Let f be a polyarrow in $\mathcal{A}$, with n inputs and m outputs; let P be an acyclic, regular, prime circuit with $\operatorname{dom} P = n$ and $\operatorname{cod} P = m$. Then $\Psi_x\Psi f \simeq (P, \cdot \mapsto f)$.*

Proof. Induction on n, the number of vertices of $\Psi_x\Psi f$.

If $n = 1$ then $\Psi_x\Psi f$ is prime with its only vertex labelled by f; by Lemma 6.49 it is acyclic. Any prime differs from a regular one only by a shuffle, hence $\Psi_x\Psi f \simeq (P, \cdot \mapsto f)$.

Otherwise, suppose $n > 1$. Since $\Psi_x\Psi f = (\Gamma, \theta)$ is acyclic and connected, it has a peripheral node v, labelled by g such that

$$(\Gamma, \theta) = (v, \theta_v) \circ (\Gamma - \{v\}, \theta - \{v\}).$$

Both of these circuits are acyclic and connected, hence by Lemma 6.49 there exist g, h of $\mathcal{A}$ such that

$$\Psi_x \Psi g = (v, \theta_v)$$
$$\Psi_x \Psi h = (\Gamma - \{v\}, \theta - \{v\})$$

and hence

$$\Psi_x \Psi f = \Psi_x \Psi g \circ \Psi_x \Psi h = \Psi_x \Psi (g \circ h)$$

Since $\Psi_x \Psi h$ has $n-1$ vertices, by induction $\Psi_x \Psi h \simeq (P_h, \cdot \mapsto h)$; on the other hand, $\Psi_x \Psi g$ is prime, hence its vertex is labelled by g. The circuit $(v, \theta_v) \circ (P_h, \cdot \mapsto h)$ contracts to an acyclic prime via the maximal contraction between the two vertices, leaving a prime labelled by $g \circ h$. However $\Psi_x \Psi$ is faithful, hence $g \circ h = f$, which gives the result. □

Proposition 6.69. *Let (Γ, θ) be a circuit of $\boldsymbol{Circ}_x^+(\mathcal{A})$; it is acyclic and connected if and only if there exists a unique f in $\mathcal{A}$ such that $\Psi_x \Psi f \simeq (\Gamma, \theta)$.*

Proof. For any polyarrow f, Lemma 6.49 states that Ψf is an acyclic, connected circuit, hence $\Psi_x \Psi f$ is also acyclic and connected.

Conversely, by the previous lemma, every acyclic connected circuit is homotopy equivalent to a regular prime (P, θ), and by Lemma 6.66 this is unique. Evidently $(P, \theta) \simeq \Psi_x \Psi \theta(\cdot)$. □

If maximal contractions are not used, cycles will be introduced in the graph. Suppose that Γ contains a pair of vertices labelled by f and g such that g is k-contractible to f along a set of edges E. For any $j < k$, let $E' \subset E$ such that $|E'| = j$. Let Δ be the new circuit which results from a contraction from g to f along E', with the new vertex labelled by $f_j \circ g$. The new vertex will have $k - j$ self loops corresponding to the edges $E - E'$, hence $\Delta = Tr_{k-j}(\Delta')$ where Δ' is a skeleton obtained by cutting all these loops. On the other hand let Γ' be the skeleton of Γ obtained by cutting $E - E'$ in the same order; clearly $\Gamma' \simeq \Delta'$. By lemma 6.44 $\Gamma = \mathrm{Tr}_{k-j}(\Gamma)$, hence $\Gamma \simeq \Delta$.

6.5.3 Circuits under Homotopy

We now consider the category $\mathbf{Circ}_{\mathrm{x}}(\mathcal{A})/\simeq$, which has the same objects as $\mathbf{Circ}_{\mathrm{x}}(\mathcal{A})$, and whose arrows are the equivalence classes of arrows of $\mathbf{Circ}_{\mathrm{x}}(\mathcal{A})$ generated by homotopy equivalence. Let $Q_\simeq$ be the evident full functor

$$\mathbf{Circ}_{\mathrm{x}}(\mathcal{A}) \xrightarrow{Q_\simeq} \mathbf{Circ}_{\mathrm{x}}(\mathcal{A})/\simeq$$

and write $[f]$ for $Q_\simeq f$, the equivalence class containing f.

Lemma 6.70. *The following are immediate from the definition.*

- *If f is a simple circuit of $\boldsymbol{Circ}_x(\mathcal{A})$ then $[f]$ is a singleton.*
- *$\boldsymbol{Circ}_x(\mathcal{A})/\simeq$ is compact closed and $Q_\simeq$ preserves the compact closed structure of $\boldsymbol{Circ}_x(\mathcal{A})$.*

Lemma 6.71. *Let $[g]$ be an arrow of $\boldsymbol{Circ}_x^+(\mathcal{A})/\simeq$; then*

$$[g] = [\Psi_x \, \mathrm{Tr}^A(\Psi f)] = \mathrm{Tr}^A[\Psi_x \Psi f]$$

for some object A, and some arrow f of $\mathcal{A}$.

Proof. By Lemma 6.65, we may take g to be prime; then g has an acyclic, regular skeleton g' such that $g = \mathrm{Tr}^A(g')$. Since $Q_\simeq$ preserves trace we have $[g] = [\mathrm{Tr}^A(g')] = \mathrm{Tr}^A[g']$; by Lemma 6.69 there exists a unique $f \in \mathcal{A}$ such that $\Psi_x \Psi f = g'$, which gives the result, since Ψ_x commutes with trace. □

The map f in the above lemma is not unique for $[g]$ since, in general, there are many acyclic skeletons of a given g; however, for a given skeleton g', f is unique.

Proposition 6.72. *The composite functor $Q_\simeq \Psi_x : \boldsymbol{Circ}(\mathcal{A}) \to \boldsymbol{Circ}_x(\mathcal{A})/\simeq$ is full and faithful.*

Proof. Lemma 6.71 implies the functor is full. For the faithfulness we consider only the positive subcategories $\mathbf{Circ}^+(\mathcal{A})$ and $\mathbf{Circ}^+_{\mathrm{x}}(\mathcal{A})/\simeq$ since, as before, every hom-set is canonically isomorphic to a positive one. Suppose that $[\Psi_x f] = [\Psi_x g]$. Pick any prime $h \in [\Psi_x f]$, and any acyclic, regular skeleton h' such that $h = \mathrm{Tr}^A(h')$; there exists a unique $h'' \in \mathcal{A}$ such that $h' \simeq \Psi_x \Psi h''$. Therefore $h \simeq Tr^A(\Psi_x \Psi h'') = \Psi_x(Tr^A(\Psi h''))$. Since h'' is unique relative to this skeleton $f = Tr^A(\Psi h'')$. This argument does not depend on f, so also $g = Tr^A(\Psi h'')$, hence $f = g$. □

Since both Ψ_x and $Q_\simeq$ are the identity on objects we have:

Theorem 6.73. *$\boldsymbol{Circ}(\mathcal{A})$ and $\boldsymbol{Circ}_x(\mathcal{A})/\simeq$ are equivalent as categories.*

6.5.4 Quotients of the Free Structure

Why go to all this effort to arrive at something equivalent to the original structure? $\mathbf{Circ}_{\mathrm{x}}(\mathcal{A})/\simeq$ provides a definition of the free compact closed category on $\mathcal{A}$ which does not depend on decomposing polyarrows into generators. Each polyarrow f corresponds to the equivalence class containing the unique regular prime labelled by f. Hence, abusing notation slightly, the embedding

$$\mathcal{A} \xrightarrow{\Phi_{\mathcal{A}}} \mathbf{Circ}_{\mathrm{x}}(\mathcal{A})/\simeq$$

can be stated directly, as $A \mapsto A$ and $f \mapsto [f]$.

Let $F : \mathcal{A} \to \mathcal{B}$ be a polycategory functor which is identity on objects. The composition $\Phi_{\mathcal{B}} F$ is a symmetric monoidal functor from $\mathcal{A}$ to $\mathbf{Circ}_{\mathrm{x}}(\mathcal{B})$, hence by the universal property of $\mathbf{Circ}_{\mathrm{x}}(\mathcal{A})$ we must have a unique compact closed functor G such that

$$\begin{array}{ccc} \mathcal{A} & \xrightarrow{\Phi_{\mathcal{A}}} & \mathbf{Circ}_{\mathrm{x}}(\mathcal{A})/\simeq \\ {\scriptstyle F}\downarrow & & \vdots\,{\scriptstyle G} \\ \mathcal{B} & \xrightarrow{\Phi_{\mathcal{B}}} & \mathbf{Circ}_{\mathrm{x}}(\mathcal{B})/\simeq \end{array}$$

commutes.

The functor G is easily defined using the structure of $\mathbf{Circ}_{\mathrm{x}}(\mathcal{A})/\simeq$. Define a partial map $\mathbf{Circ}_x(F)/\simeq$ between the arrows of $\mathbf{Circ}_{\mathrm{x}}(\mathcal{A})/\simeq$ and $\mathbf{Circ}_{\mathrm{x}}(\mathcal{B})/\simeq$ by

$$\mathbf{Circ}_x(F)/\simeq ([f]) = [Ff]$$

whenever f is an acyclic prime. This is well defined since, by Lemma 6.66, if $[f]$ contains such an element it is unique.

Proposition 6.74. ***Circ***$_x(F)/\simeq$ *extends to the required compact closed functor* G *above.*

Proof. Take $\mathbf{Circ}_x(F)/\simeq$ to be identity on objects. The requirement to preserve the compact closed structure fixes the value of $\mathbf{Circ}_x(F)/\simeq$ on the rest of the arrows, since the acyclic primes generate the other arrows. To show that it preserves composition, consider two arrows in $\mathbf{Circ}^+_{\mathrm{x}}(\mathcal{A})/\simeq$, f and g. For a general arrow

$$f = [\mathrm{Tr}^A(\bigotimes_i f_i)]$$

with each of the f_i acyclic, prime and regular. Hence

$$\begin{aligned}\mathbf{Circ}_x(F)/\simeq (f) \circ \mathbf{Circ}_x(F)/\simeq (g) &= [\mathrm{Tr}^A(\bigotimes_i Ff_i)] \circ [\mathrm{Tr}^B(\bigotimes_j Fg_j)] \\ &= \mathrm{Tr}^{A\otimes B}[(\bigotimes_i Ff_i) \circ (\bigotimes_j Fg_j)]\end{aligned}$$

The circuit $(\otimes_i Ff_i) \circ (\otimes_j Fg_j)$ is necessarily acyclic, hence each of its connected components is the image of a well defined composition of polyarrows $Ff_{i_1} \circ Fg_{j_2} \circ \cdots \circ Ff_{i_n} = F(f_{i_1} \circ g_{j_2} \circ \cdots \circ f_{i_n}) = Fh_k$.

$$\begin{aligned}\mathrm{Tr}^{A\otimes B}[(\bigotimes_i Ff_i) \circ (\bigotimes_j Fg_j)] &= \mathrm{Tr}^{A\otimes B}[(\bigotimes_k Fh_k)] \\ &= \mathbf{Circ}_x(F)/\simeq (f \circ g).\end{aligned}$$

□

Remark. The restriction that F be identity on objects can be relaxed to the requirement that F be injective on objects without causing any problems.

A particular case of interest is when $\mathcal{B}$ is obtained from the freely generated $\mathcal{A}$ by imposing a set of equations between certain polyarrows. If $Ff = Fg$ the arrows of the free compact closed category are obtained by merging the equivalence classes generated by these arrows, hence the decision problem for $\mathbf{Circ}_{\mathrm{x}}(\mathcal{B})$ is essentially reduced to that for $\mathcal{B}$.

6.6 Generalised Proof-nets

We now define a system of two-sided proof-nets constructed over the generators of a polycategory with multicut $\mathcal{A}$. The resulting system of proof-nets will be denoted PN($\mathcal{A}$) .

Definition 6.75. A PN($\mathcal{A}$) *proof-net* is a finite directed graph with edges labelled by **mCQL** formulae (see Section 4.1) . The graph is constructed by composing the following links, while respecting the labelling on the incoming and outgoing edges.

Premise: No incoming edges; one outgoing edge. The edge is labelled with an arbitrary formula and the link is unlabelled.

Conclusion: One incoming edge; no outgoing edges. The edge is labelled with an arbitrary formula and the link is unlabelled.

Unit: No incoming edges; two outgoing edges. The first outgoing edge is labelled X^*, the other, X, for some formula X. The link itself is labelled by η.

Counit: Two incoming edges; no outgoing edges. Each counit is labelled by ϵ and its incoming edges are labelled by X and X^* for an arbitrary formula X.

Tensor: Two incoming edges labelled X and Y; one outgoing edge labelled $X \otimes Y$.

Cotensor: One incoming edge labelled $X \otimes Y$; two outgoing edges labelled X and Y.

Circle: No incoming or outgoing edges; a circle is a closed loop labelled by a formula.

Axiom: Each polyarrow $f : \otimes_i A_i \to \otimes_j B_j$ in $\mathrm{Arr}_{\mathcal{A}}$ defines a link labelled by f. Its n incoming edges are labelled by $A_1, \ldots, A_n$ and its m outgoing edges are labelled by $B_1, \ldots, B_m$.

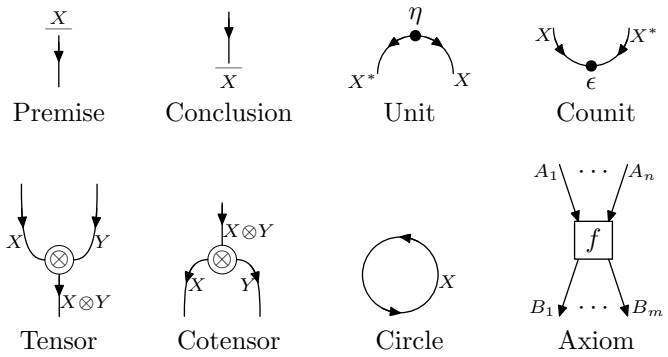


Figure 6.10: Links for PN($\mathcal{A}$) Proof-nets

A proof-net is *oriented* such that edges enter the node from the top, and exit from the bottom. This implies that any premise or unit link is above the links they are connected to and, conversely, any conclusion or counit links are below the links they are connected to.

The order of premises and conclusions is significant, and the *type* of a proof-net is the pair (Γ, Δ) of lists of formulae determined by the premises and conclusions respectively. Usually this will be written as a sequent $\Gamma \vdash \Delta$. A premise or conclusion link is called *atomic* if the formula labelling it is a literal; a proof-net is called atomic if all its premises and conclusions are atomic.

Proof-nets are permitted to be disconnected or cyclic, when considered as directed or undirected graphs. In particular, an edge may leave a link and return as an input to the same link, although the labelling on edges will prohibit this

for all except axiom links. If a proof-net is directed-acyclic then it is called *process-like.*

Definition 6.76 (β-reduction)**.** Let ν, μ be proof-nets; define a one step reduction relation on proof-nets R_β such that $\nu \, R_\beta \, \mu$ if ν can be rewritten to μ by one of the local rewrite rules shown in Figures 6.11 and 6.12. Let $\xrightarrow[\beta]{}$ be the transitive, reflexive closure of R_β and let $=_\beta$ be the symmetric closure of $\xrightarrow[\beta]{}$.

Lemma 6.77 (Subject Reduction)**.** *Suppose that ν is a proof-net with type (Γ, Δ) and $\nu \xrightarrow[\beta]{} \mu$; then μ also has type (Γ, Δ).*

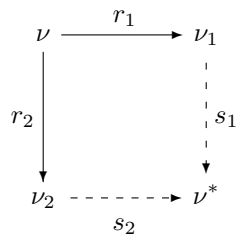
Proof. No rewrites change the premises or conclusions, hence the type is unchanged by β-reduction. □

Theorem 6.78 (Termination)**.** *There is no infinite sequence of β-reductions.*

Proof. Before the main part of the proof we can dispose of one of the rewrite rules. Notice that the right hand side of the circle reversal rule cannot be rewritten further, since the label is a positive literal. Since our proof-nets are finite, there can only be finitely many circle reversal rewrites. Hence this rule has no effect on termination, so we can safely ignore it.

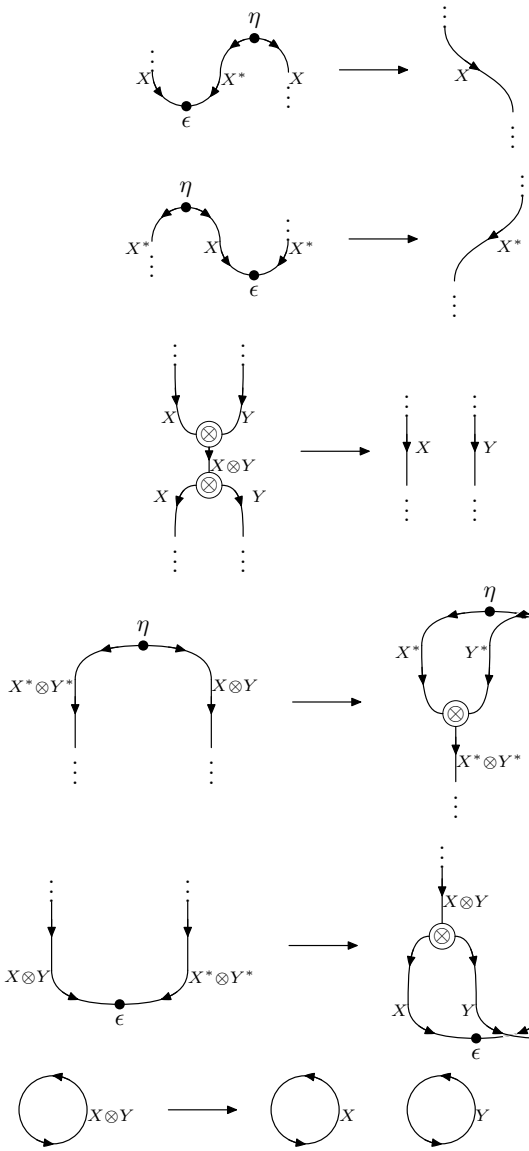
Divide the edges of a proof-net ν into two classes: the "bad" edges are those which connect two tensor links, two cotensor links, a premise to cotensor, or a tensor to conclusion; the "good" edges are all the rest. Let $r(\nu)$ be the number of occurrences of the the symbol $\otimes$ as the label on a good edge of ν. Let $d(\nu)$ be the number of links in ν. Define an ordering on proof-nets by by $\nu < \mu$ if $r(\nu) < r(\mu)$, or if $r(\nu) = r(\mu)$ and $d(\nu) < d(\mu)$. An inspection of the rewrite rules reveals that if $\nu \xrightarrow[\beta]{} \mu$ then necessarily $\nu > \mu$. Since both formulae and proof-nets are finite, there can be no infinite descending chain in this order, and hence every rewrite sequence terminates. □

Theorem 6.79 (Local Confluence)**.** *If a proof-net ν β-reduces to ν_1 and ν_2 by different rewrites r_1, r_2, then there exist sequences of rewrites s_1, s_2 such that*



Proof. Since all the rewrites are local, it suffices to show that any pair of overlapping rewrites which diverge can be unified. There are 15 such divergences.

Firstly, the rules $\eta\epsilon$-elim 1 and 2 may conflict in the three ways shown in Figure 6.13. The first two divergences unify vacuously; the third case requires an easy induction on the structure of the formula X. If X atomic, then the circle labelled by X is already normal, and the circle labelled by X^* rewrites to this by the circle reversal rule. Otherwise if $X = Y \otimes Z$, the circle then rewrites via the circle split rule to a pair of circles labelled by Y and Z which,



If A is an atom then:

Figure 6.11: Rewrite Rules for PN($\mathcal{A}$), I

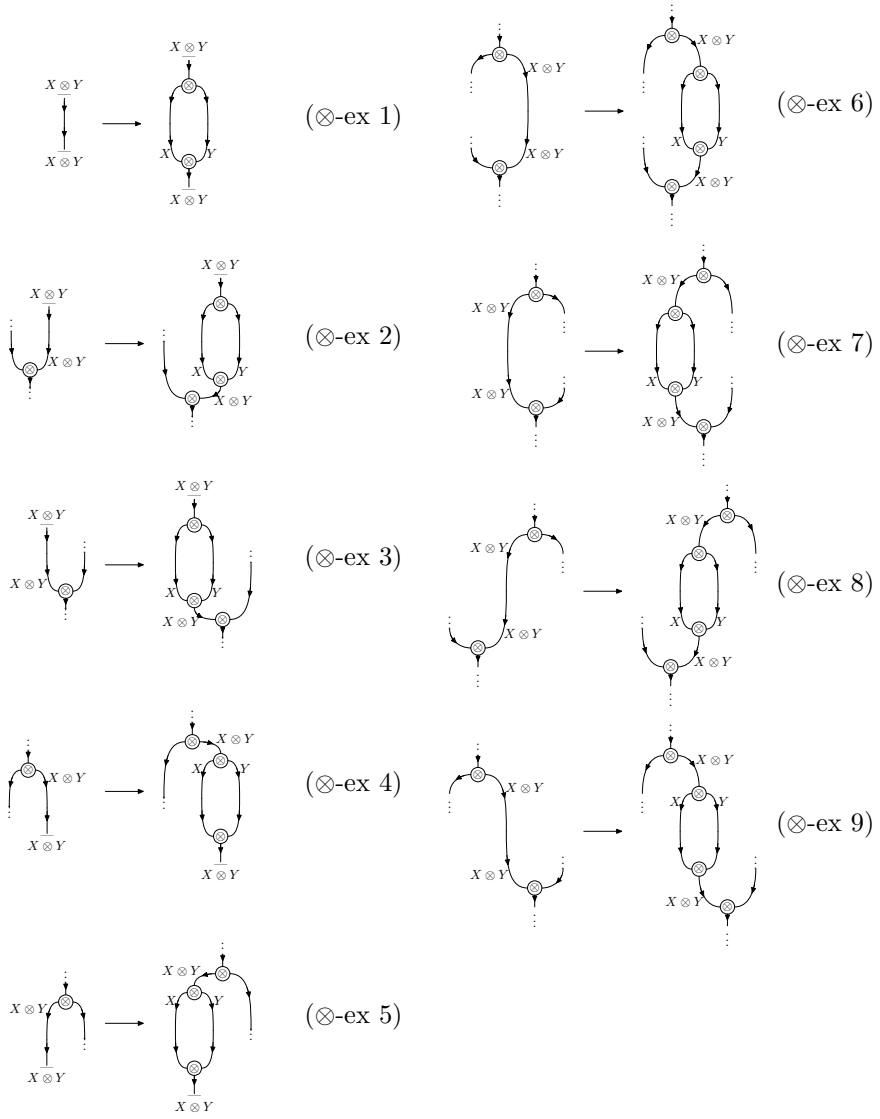


Figure 6.12: Rewrite Rules for PN($\mathcal{A}$). II: Tensor Expansions

by induction hypothesis, are β-equal with the pair of circles labelled by Y^* and Z^*.

Secondly, $\eta\epsilon$-elim 1 and 2 may both clash with the rules η-ex and ϵ-ex. These four cases are all essentially the same, so we treat only the divergence between $\eta\epsilon$-elim 1 and η-ex. Suppose that we have

$$\pi_1 \xleftarrow{\eta\epsilon\text{-elim 1}} \pi \xrightarrow{\eta\text{-ex}} \pi_2$$

as shown in Figure 6.14. The resolution depends on the adjacent links L_1 and L_2. There are four legal choices for L_1: premise, unit, tensor and cotensor. Dually, the four possibilities for L_2 are conclusion, counit, tensor and cotensor, giving a total of 16 pairings.

- If L_1 is a tensor link then π_2 rewrites to π_1 by $\otimes$-elim.
- If L_1 is a unit link, π_1, π_2 rewrite to π_1', π_2' by η-ex whereupon π_2' rewrites to π_1' by $\otimes$-elim.
- If L_1 is a premise and L_2 is a conclusion then π_1 rewrites to π_2 by $\otimes$-ex 1.
- If L_1 is a premise and L_2 is a tensor, then π_1 rewrites to π_2 by $\otimes$-ex 2 or 3, depending on which input of L_2 is used.
- If L_1 is a cotensor and L_2 a tensor, then π_1 rewrites to π_2 via one of $\otimes$-ex 6–9 depending on the arrangement of the inputs and outputs.

These cases are enumerated in Table 6.1 and their unifications are shown in Figures 6.15, 6.16 and 6.17.

Finally, the rule for $\otimes$-elim may conflict with each the expansions $\otimes$-ex 2–9. All of these are essentially the same so we treat only the case of $\otimes$-ex 8, shown in Figure 6.18. Suppose

$$\pi_1 \xleftarrow{\otimes\text{-elim}} \pi \xrightarrow{\otimes\text{-ex 8}} \xrightarrow{\otimes\text{-elim}} \pi_2$$

There are four cases, based on what the link labelled L is.

- If L is a cotensor, π_2 rewrites to π_1 by $\otimes$-elim;
- if L is a tensor, π_1 rewrites to π_2 by $\otimes$-ex 7 or 8;
- if L is a conclusion, π_1 rewrites to π_2 by $\otimes$-ex 5;
- if L is a counit, π_1, π_2 rewrite to π_1', π_2' by ϵ-ex whereupon π_2' rewrites to π_1' by $\otimes$-elim.

These four cases are shown in Figure 6.19.

Since all divergent rewrites can be unified, PN($\mathcal{A}$) is locally confluent under β-reduction. □

Theorem 6.80 (Strong Normalisation)**.** *β-reduction for proof-nets is strongly normalising.*

Proof. Since β-reduction is confluent, each proof-net has a unique normal form; since it is terminating, every rewrite sequence must arrive at the normal form. □

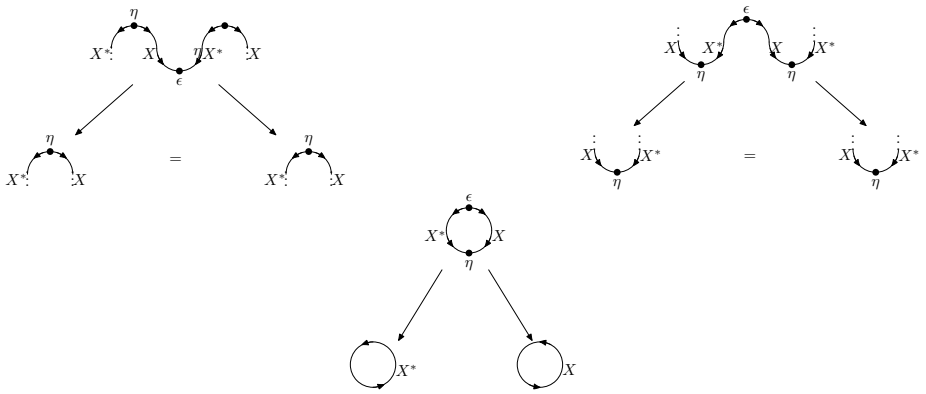


Figure 6.13: Unifying $\eta\epsilon$-elim Divergences

L_1 / L_2	Premise $\frac{X}{}$	Unit η : X^*, X	Cotensor $X\otimes Y$: X, Y	Tensor X, Y : $X\otimes Y$
Conclusion $\frac{}{X}$	1	2	3	4
Counit ϵ : X, X^*	Dual to 2	5	6	7
Tensor X, Y : $X\otimes Y$	Dual to 3	Dual to 6	8	9
Cotensor $X\otimes Y$: X, Y	Dual to 4	Dual to 7	Dual to 9	10

Table 6.1: Enumeration of cases for $\eta\epsilon$-elim / η-ex divergence

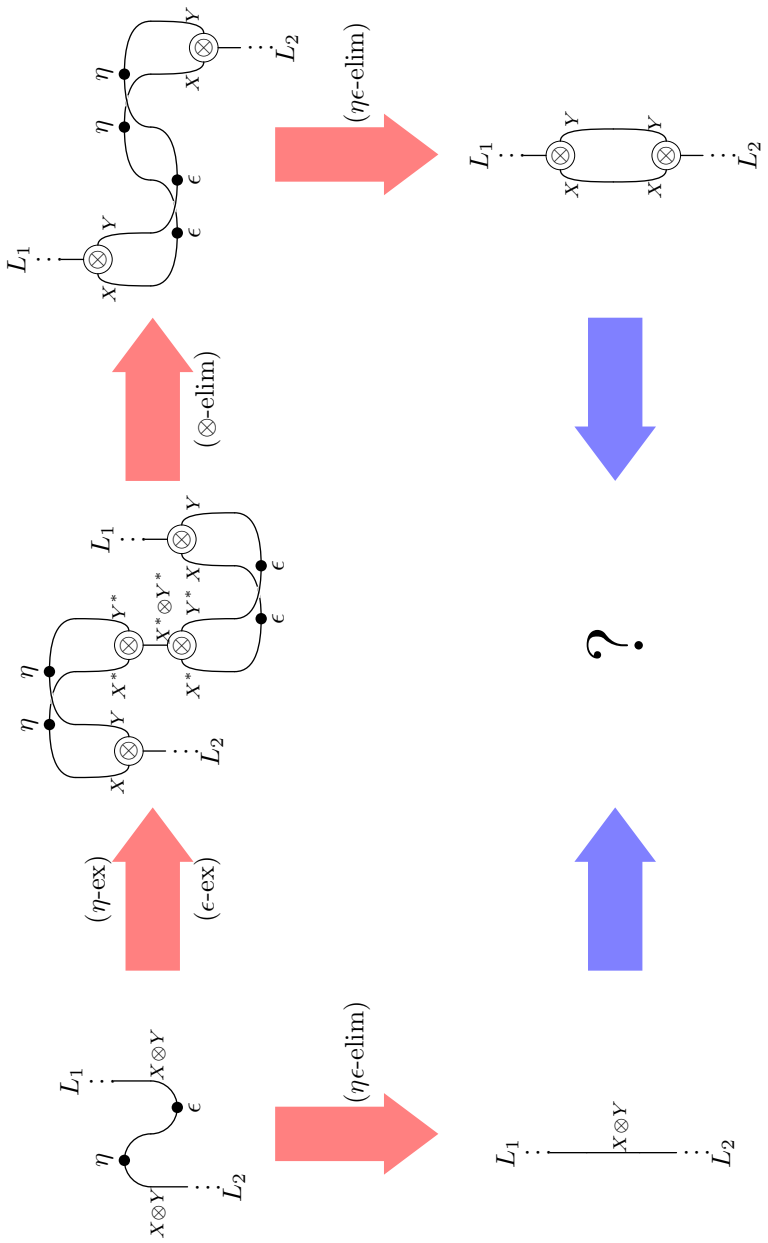


Figure 6.14: Divergence between $\eta\epsilon$-elim 1 and η-ex

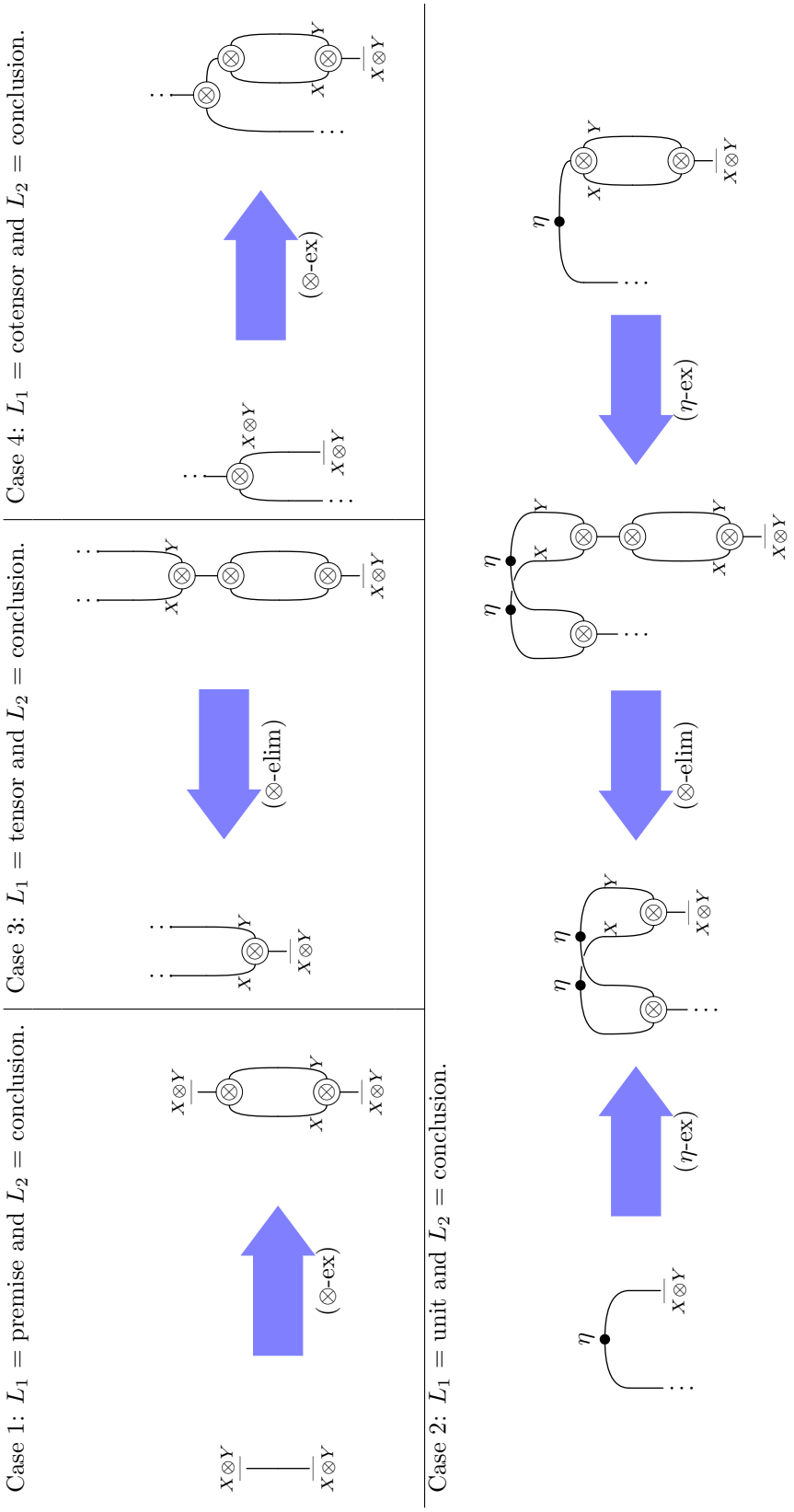


Figure 6.15: Unifying $\eta\epsilon$-elim and η-ex divergences, cases 1-4.

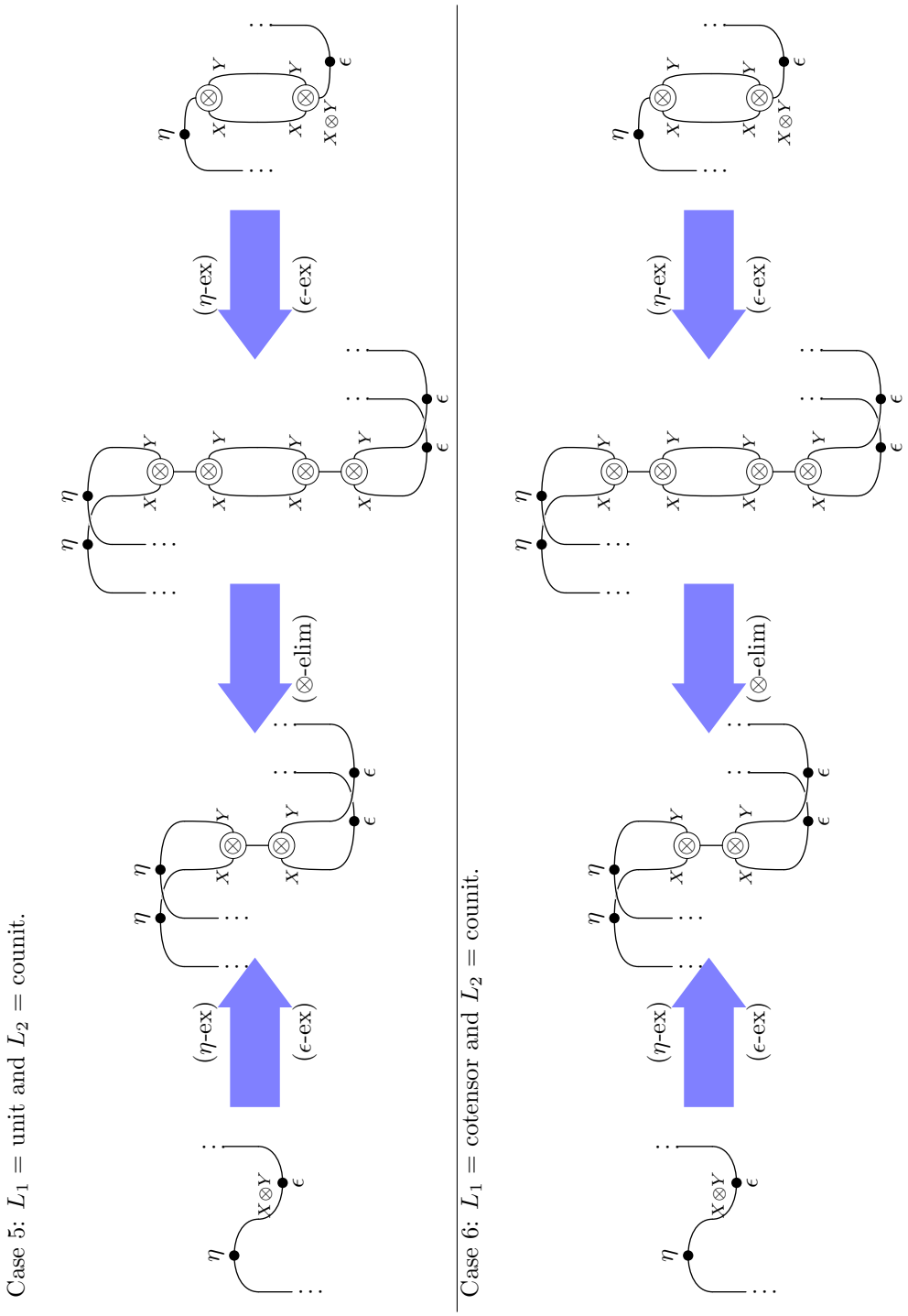
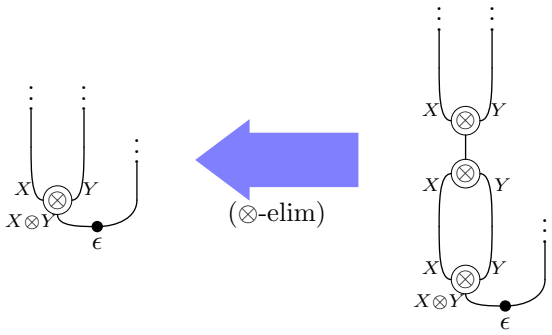
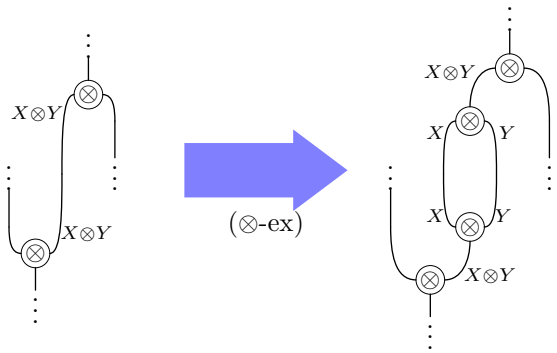


Figure 6.16: Unifying $\eta\epsilon$-elim and η-ex divergences, cases 5-6.

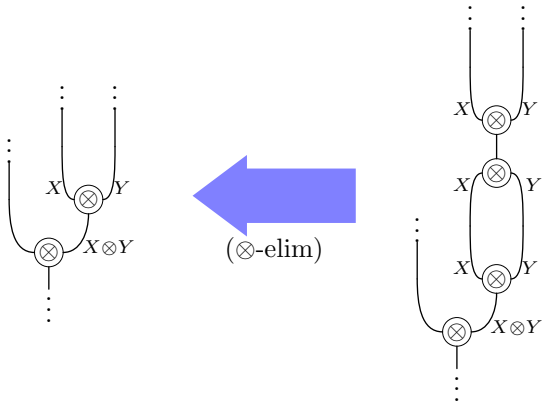
Case 7: L_1 = tensor and L_2 = counit.



Case 8: L_1 = cotensor and L_2 = tensor.



Case 9: L_1 = tensor and L_2 = tensor.



Case 10: L_1 = tensor and L_2 = cotensor.

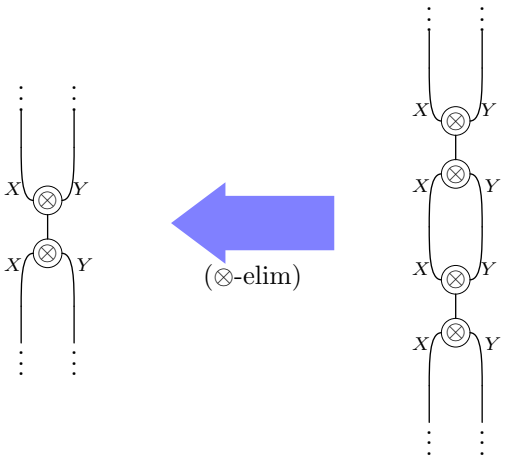


Figure 6.17: Unifying $\eta\epsilon$-elim and η-ex divergences, cases 7-10.

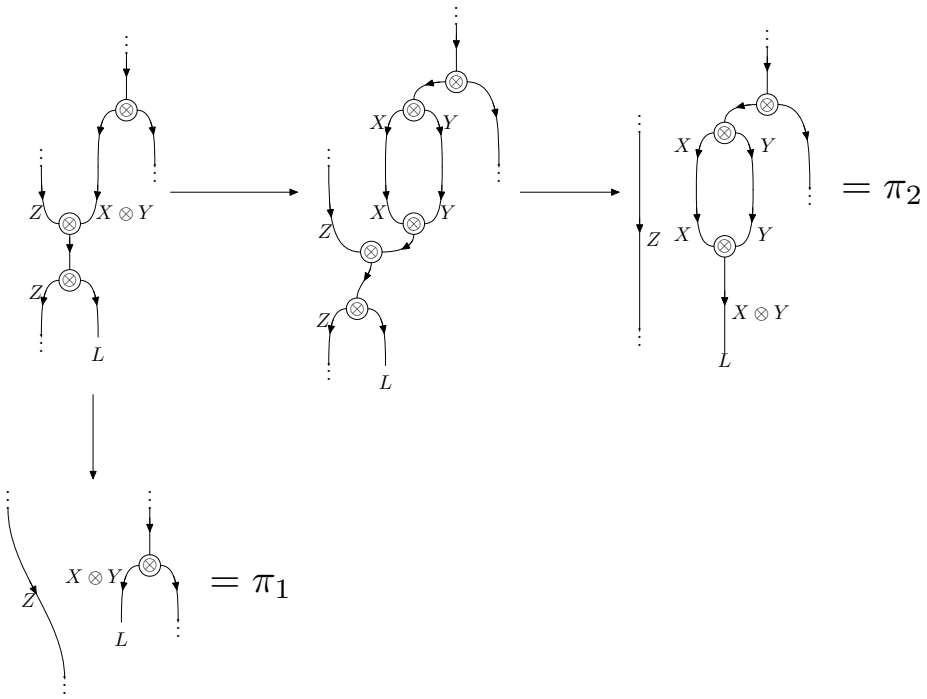


Figure 6.18: Divergence between ⊗-elim and ⊗-ex 8

Having established the existence of β-normal proof-nets, we now characterise them intrinsically. Recall that for multiplicative linear logic proof nets[Gir87b], the structure of a cut free proof can be separated into the axiom structure and the connective structure. The following lemmas give a similar result, pushing the connectives to the outside of the proof-net.

Lemma 6.81. *Let ν be a normal proof-net, and suppose x is a link in ν.*

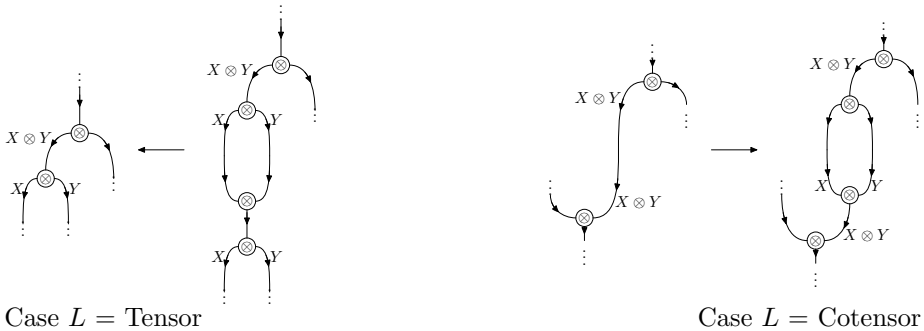
- *If x is a tensor link, all links below x are tensors or conclusions.*
- *If x is a cotensor link, all links above x are contensors or premises.*

Proof. Let x be a tensor link. Its outgoing edge is labelled by some formula $X \otimes Y$; suppose there is a link below it, called x'.

- If x' is a cotensor then rewrite rule ⊗-elim applies, hence ν is not normal.
- If x' is a counit then it is labelled by $X \otimes Y$, hence rewrite rule ϵ-ex applies and ν is not normal.
- If x' is an axiom, it has an incoming edge labelled by a non-atomic formula, which contradicts the definition of axiom link.

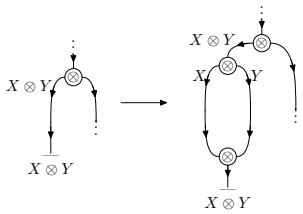
Hence x' is either a tensor or a conclusion. If it is a conclusion then the hypothesis is satisfied. If x' is a tensor, then by induction all the links below x' are also tensors or conclusions. Hence if ν is normal, then all links below a tensor link are also tensors or conclusions.

The case when x is a cotensor is exactly dual. □

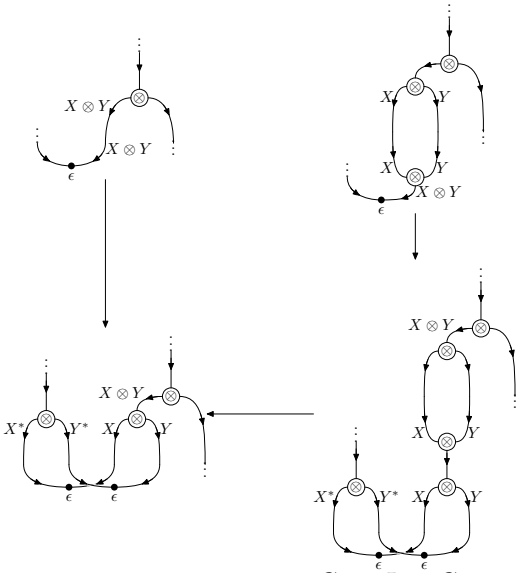


Case L = Tensor

Case L = Cotensor



Case L = Conclusion



Case L = Counit

Figure 6.19: Unifying ⊗-elim/⊗-ex divergences

Corollary 6.82. *Any normal proof-net ν can be formed from a normal atomic net ν' by adding tensor links to its conclusions and cotensors to its premises.*

Corollary 6.83. *All the edges of a normal atomic proof-net are labelled by literals.*

Proposition 6.84. *An atomic proof-net is normal if and only if all its edges are labelled by atomic formulae and no unit link is connected to a counit.*

Proof. If ν is normal, Corollary 6.83 gives that all its edges' labels are atomic; by its normality no unit is connected to a counit since otherwise rewrite $\eta\epsilon$-elim 1 or 2 would apply.

Conversely, suppose that ν is atomic, such that all its edges are labelled by literals, and none of its unit links are connected to counits. Since all its edges are labelled by literals, none of η-ex, ϵ-ex, circle split, or $\otimes$-ex 1 can apply. For the same reason it contains no tensor or cotensor links, hence rewrites $\otimes$-elim and $\otimes$-ex 2–9 do not apply. By hypothesis, no unit is connected to a counit, hence rewrites $\eta\epsilon$-elim 1 and 2 cannot apply. Since, no rewrites are possible, ν is in its normal form. □

Definition 6.85. Let π, ν be proof-nets of types $\Gamma \vdash \Delta$ and $\Gamma' \vdash \Delta'$. Define their tensor product $\pi \otimes \nu$ to be the proof-net formed by their juxtaposition, with type $\Gamma, \Gamma' \vdash \Delta, \Delta'$.

Suppose further that $\Gamma' = \Gamma_1, X, \Gamma_2$ and $\Delta = \Delta_1, X, \Delta_2$; then the composition of π and ν along X is formed by identifying the X-conclusion of π with the X-premise of ν to produce a proof-net of type $\Gamma_1, \Gamma, \Gamma_2 \vdash \Delta_1, \Delta, \Delta_2$.

Proposition 6.86. $\mathrm{PN}(\mathcal{A})/=_\beta$ *is a polycategory.*

Proof. The objects of the polycategory are the formulae of $\mathrm{PN}(\mathcal{A})$; the polyarrows are the proof-nets. Routine work verifies that the required equations are satisfied. □

Indeed, one can show that $\mathrm{PN}(\mathcal{A})$, with the tensor defined above, is actually a polycategory with self-dual tensor. See [CS97] for definitions.

6.6.1 $\mathrm{PN}(\mathcal{A})$ is equivalent to $\mathbf{Circ}(\mathcal{A})$

The normal atomic proof-nets are very closely related to the $\mathcal{A}$-labellable circuits. A simple formal transformation produces a circuit from each such proof-net, and vice-versa. These transformations define a pair of functors

$$\mathbf{Circ}(\mathcal{A}) \underset{U}{\overset{F}{\rightleftarrows}} \mathrm{PN}(\mathcal{A})$$

which form an equivalence of categories. We approach the proof via the special case of normal atomic proof-nets.

Lemma 6.87. *Suppose ν is an atomic normal proof-net; suppose e is an edge in ν labelled by a negative literal. One of the following holds:*

- *e connects a premise to a conclusion;*
- *e connects a premise to a counit link;*

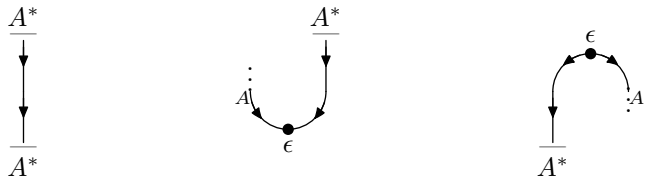


Figure 6.20: Negative Edges

- *e connects a unit link to a conclusion.*

Proof. By Lemma 6.81, ν contains no tensor or cotensor links; axioms cannot have negative edges, therefore e must connect either a premise, unit, counit or conclusion. Since the proof-net is normal, e cannot join a unit to a counit by the preceding lemma. Since an edge cannot be incoming or outgoing at both endpoints the pairings unit/unit, counit/counit, premise/premise, conclusion/conclusion, conclusion/counit and premise/unit are excluded. This leaves the three possibilities claimed. These can occur validly in a normal proof-net as shown by Fig. 6.20. □

Suppose that $\pi : \Gamma \to \Delta$ is normal and atomic; then π can be rewritten to produce an $\mathcal{A}$-labelled circuit $c(\pi) : \bigotimes \Gamma \to \bigotimes \Delta$ by the following procedure.

1. The premises and conclusions of π become the boundary nodes of $c(\pi)$; the premises form $\operatorname{dom} c(\pi)$ and the conclusions $\operatorname{cod} c(\pi)$. They are labelled by the edges formulae and signed according to whether the atom is positive or negative.

2. For all edges e labelled by a negative literal A^*, reverse e's direction, and change its labelling to A. This guarantees that negatively signed nodes in the codomain have incoming edges, and vice versa.

3. Erase every unit and counit node, merging their incident edges, which are now pointing in the same direction.

4. The remaining links of π must all be axioms links. These become the internal nodes of $c(\pi)$. At each node x, the ordering on $\operatorname{in}(x)$ and $\operatorname{out}(x)$ is simply that of the components of the domain and codomain of the arrow (in $\mathcal{A}$) which labels that node.

Lemma 6.87 guarantees that $c(n)$ really is a circuit. There is a dual procedure, taking a circuit $f : \bigotimes_i A_i \to \bigotimes_j B_j$ to a normal atomic proof-net.

1. The nodes in $\operatorname{dom} f$ become premises; those of $\operatorname{cod} f$, conclusions.

2. If e is an edge, labelled by A, going from some node n to a premise p, replace e with a counit-link whose incoming edges are from e and p, labelled by A and A^* respectively.

3. If e is an edge, labelled by B, going to some node n from a conclusion c, replace e with a unit-link whose outgoing edges go to e and c and are labelled by B and B^* respectively.

4. The interior nodes of f become axiom links, each determined by the label on the corresponding node.

This defines a proof-net $p(f) : A_1, \ldots, A_n \vdash B_1, \ldots B_m$, which by Prop. 6.84 is normal. The two procedures are mutually inverse, which leads to the following characterisation result.

Theorem 6.88. *Let π be a normal proof-net; then π is completely determined by its type and an $\mathcal{A}$-labellable circuit.*

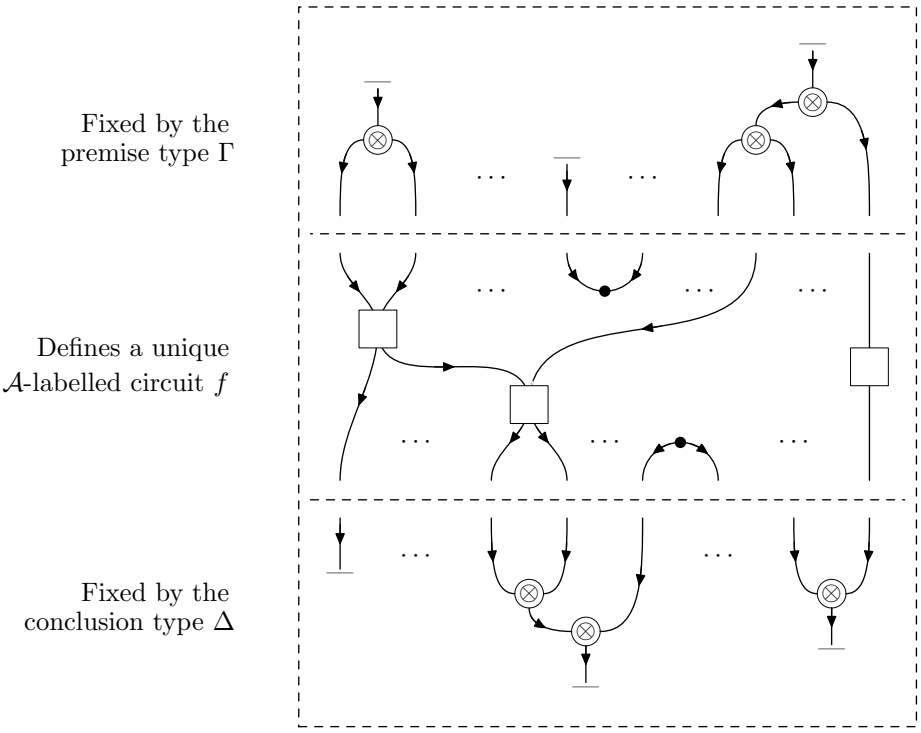


Figure 6.21: Normal Proof-net decomposition

Proof. Suppose that π has type $\Gamma \vdash \Delta$. Given a formal list of formula Γ, let Γ^- be the list of literals produced by replacing every occurrence of $\otimes$ by a comma. By Lemma 6.81, π can be decomposed into three layers: on top π_Γ of type $\Gamma \vdash \Gamma^-$, consisting only of cotensor links; the middle $\pi^- : \Gamma^- \vdash \Delta^-$ which is both normal and atomic; and at the bottom $\pi_\Delta : \Delta^- \vdash \Delta$ consisting only of cotensors. The layers π_Γ and π_Δ are uniquely determined by Γ and Δ, while π^- is uniquely determined by the circuit $c(\pi^-)$. □

Justified by this theorem we write $\pi \sim (\Gamma \vdash \Delta, f)$ for any normal π. The required functors F and U are now easily defined.

For each A of $\mathrm{PN}(\mathcal{A})$, let UA be the positively signed singleton, labelled by A; then define $U(A^*) = (UA)^*$ and $U(X \otimes Y) = UX \otimes UY$. Let $\pi \xrightarrow[\beta]{} \nu \sim (\Gamma \vdash \Delta, f)$ where ν is normal; then define $U\pi = f$.

To map $\mathbf{Circ}(\mathcal{A})$ into $\mathrm{PN}(\mathcal{A})$, let f be a circuit; then let Ff be the proof-net obtained from $p(f)$ by adding tensor links to all the conclusions (bracketed to the left) and, similarly, cotensors to all the premises.

Theorem 6.89. *The 4-tuple* $(\boldsymbol{Circ}(\mathcal{A}), \mathrm{PN}(\mathcal{A}), F, U)$ *is an equivalence of categories.*

Proof. Obviously, from the construction of U and F, we have $UF = \mathrm{Id}$. On the other hand, a proof-net $\pi : X \to Y$ only differs from $FU\pi : FUX \to FUY$ by the associativity of the tensor, hence $\mathrm{Id} \cong FU$. □

6.7 Relations and Rewriting

Let $\mathcal{A}$ be a polycategory and suppose that $\mathcal{E}$ is a binary relation on the arrows of $\mathcal{A}$ such that if $(f, g) \in \mathcal{E}$ then $\mathrm{dom}\, f = \mathrm{dom}\, g$ and $\mathrm{cod}\, f = \mathrm{cod}\, g$. Let $\mathcal{R}$ be the least congruence with respect to composition and symmetry containing $\mathcal{E}$, modulo the equations for compact symmetric polycategories. We can view $\mathcal{R}$ as a rewriting system on the arrows of $\mathcal{A}$.

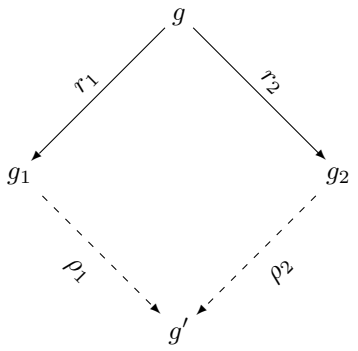
Consider a pair of arrows $f, g \in \mathcal{A}(A, B)$. Their images in $\mathbf{Circ}(\mathcal{A})$, Ψf and Ψg, have the same boundary. Hence, if the open graph $\Psi f - \partial \Psi f$ occurs as a subgraph of some circuit Γ, replacing it with the interior of Ψg yields a valid circuit Γ', which is also $\mathcal{A}$-labelled. Hence the rewriting system $\mathcal{R}$ lifts from $\mathcal{A}$ to $\mathbf{Circ}(\mathcal{A})$.

Proposition 6.90. *Let* $\mathcal{R}$ *be as described above, and let* $\hat{\mathcal{R}}$ *be the induced rewriting on* $\boldsymbol{Circ}(\mathcal{A})$*. Then, if* $\mathcal{R}$ *is locally confluent then so is* $\hat{\mathcal{R}}$*.*

Proof. It suffices to consider $\mathbf{Circ}(\mathcal{A})^+$. Let Γ be a circuit such that there are divergent rewrites

$$\Gamma_1 \xleftarrow{\hat{r_1}} \Gamma \xrightarrow{\hat{r_2}} \Gamma_2$$

Suppose that γ_1 and γ_2 are the domains of $\hat{r_1}, \hat{r_2}$ respectively. Since $\hat{r_1}, \hat{r_2}$ correspond to rewrites $g_1 \xrightarrow{r_1}$ and $g_2 \xrightarrow{r_2}$ in $\mathcal{A}$, γ_1, γ_2 are necessarily acyclic. Any minimal subgraph of Γ containing both γ_1 and γ_2 is also acyclic, and hence it is the image of some arrow g in $\mathcal{A}$, to which the rewrites r_1, r_2 may be applied. $\mathcal{R}$ is locally confluent so we have



and both the paths ρ_1, ρ_2 lift to $\mathbf{Circ}(\mathcal{A})$. Hence $\hat{\mathcal{R}}$ is locally confluent. □

Local confluence suffices for confluence, but unfortunately the bargains end here: neither termination nor weak normalisation lift to circuits. It is easy to manufacture examples where $\mathcal{R}$ is strongly normalising, but $\hat{\mathcal{R}}$ has infinitely many non-normalisable terms, due to cycles induced by the trace.

Lemma 6.91. *Let $\mathcal{R}$ be a terminating rewrite system on the polycategory $\mathcal{A}$; if $f \in \boldsymbol{Circ}(\mathcal{A})$ is acyclic then no infinite rewrite sequence in $\hat{\mathcal{R}}$ contains f.*

Proof. Since f is acyclic, it is isomorphic as a graph to some polyarrow, hence any infinite rewrite sequence including f would contradict the termination property of $\mathcal{R}$. □

Call the binary relation $\mathcal{E}$ *strictly reducing* if whenever (f, g) is in $\mathcal{E}$ then g contains strictly fewer generators than f. Note that if $\mathcal{E}$ is strictly reducing then $\mathcal{R}$ is *terminating*: it contains no infinite rewrite sequences. If any rewrite system is terminating and confluent it is strongly normalising.

Proposition 6.92. *If $\mathcal{E}$ is strictly reducing then $\hat{\mathcal{R}}$ is terminating.*

Remark. It should be possible to weaken the condition of strict reduction: all that is really needed is that normal forms be invariant under cyclic (partial) permutation.

Let $\sim$ be the equivalence relation on the arrows of $\mathcal{A}$ generated by $\mathcal{R}$, and let $\hat{\sim}$ be the equivalence relation on the arrows of $\mathbf{Circ}(\mathcal{A})$ generated by $\hat{\mathcal{R}}$.

Proposition 6.93. *The quotient category $\boldsymbol{Circ}(\mathcal{A})/\hat{\sim}$ is compact closed.*

Proof. The compact closed structure is given by simple circuits. Since $\mathcal{E}$ is defined over the generators, these are unchanged. □

If $f \sim g$ then $\Psi f \hat{\sim} \Psi g$, hence the embedding $\Psi/\sim$:

$$\begin{array}{ccc} \mathcal{A} & \xrightarrow{\Psi} & \mathbf{Circ}(\mathcal{A}) \\ {\scriptstyle E}\downarrow & & \downarrow{\scriptstyle \hat{E}} \\ \mathcal{A}/\sim & \xrightarrow[\Psi/\sim]{} & \mathbf{Circ}(\mathcal{A})/\hat{\sim} \end{array}$$

is well defined.

Proposition 6.94. *Let $\mathcal{C}$ be a compact closed category and $G : \mathcal{A} \to \mathcal{A}$ a strong monoidal functor such that $G \cong G^\natural \Psi$. If G factors through $\mathcal{A}/\sim$ then $G^\natural$ factors through $\boldsymbol{Circ}(\mathcal{A})/\hat{\sim}$.*

Proof. Note that

$$\begin{array}{rcl} \Psi f \hat{\sim} \Psi g & \Rightarrow & f \sim g \\ & \Rightarrow & Gf = Gg \\ & \Rightarrow & G^\natural \Psi f = G^\natural \Psi g. \end{array}$$

Since both $\hat{\sim}$ and $G^\natural$ preserve the compact closed structure we have $f' \hat{\sim} g' \Rightarrow G^\natural f' = G^\natural g'$. □

6.8 Example: Proving No-cloning

In Section 1.1 we saw the proof of the no cloning theorem for quantum states in the Hilbert space formalism. As a simple example of the circuit representation, we repeat that proof in the abstract setting. In Hilbert spaces, the proof boiled down to the claim that the semi-ring of scalars — in that case the complex numbers — contained no idempotents except 1 and 0; in the circuit setting the proof also depends on the properties of the scalars.

Let $\mathcal{A}$ be a polycategory, freely generated from some collection of objects and polyarrows, such that $\mathcal{A}$ contains some "states"

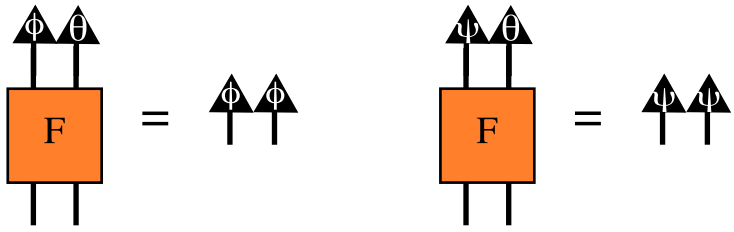
$$\psi : - \to A \qquad \phi : - \to A \qquad \theta : - \to A$$

We will represent these diagrammatically as:

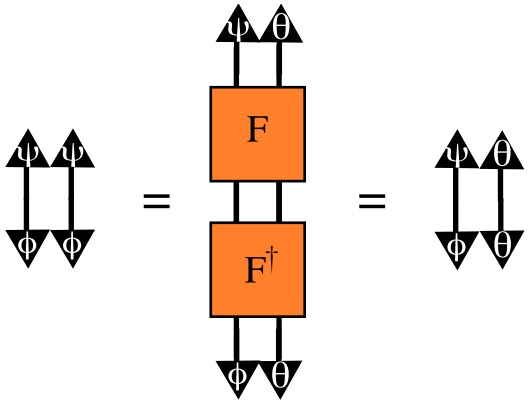
Now suppose that $\mathbf{Circ}(\mathcal{A})$ contains a unitary cloning map, $F : A \otimes A \to A \otimes A$. By cloning map, we intend that, given θ as an ancilla input, F will produce two copies of its other input as output. That is, we have the equations

$$F \circ (\phi \otimes \theta) = \phi \otimes \phi,$$
$$F \circ (\psi \otimes \theta) = \psi \otimes \psi,$$

which can be expressed as the diagrammatic equations:



To complete the proof, we simply compose the second equation with the adjoint of the first, and use the unitarity of F:



Since $\mathbf{Circ}(\mathcal{A})$ is freely generated this equation does not hold, which gives a contradiction to the supposed cloning property of F.

Recall that $\phi^\dagger \circ \psi$ is the scalar product $\langle \phi \mid \psi \rangle$, and so the original proof is reclaimed by assuming $\langle \theta \mid \theta \rangle = 1_I$, but this is not necessary.

It is worth pausing to consider what is actually proved by the above argument. The construction above is a re-enactment of the standard proof in the abstract setting; in particular it is not a derivation of no-cloning from logical or structural reasons. Rather, the proof shows that if, for contingent reasons, the cloning map exists then it implies an equation among the scalars. Since $\mathbf{Circ}(\mathcal{A})$ is freely generated this equation cannot hold, therefore there is no cloning map in $\mathbf{Circ}(\mathcal{A})$. Of course, it easy to construct an interpretation of $\mathbf{Circ}(\mathcal{A})$ into **FDHilb** where this equality is indeed violated: simply interpret ψ and ϕ as distinct, non-orthogonal states.

There are, however, models of $\mathbf{Circ}(\mathcal{A})$ where these equations do hold. For example, in **Rel** the only scalars are the empty map and the identity, and hence the equation $\langle \phi \mid \psi \rangle^2 = \langle \phi \mid \psi \rangle \langle \theta \mid \theta \rangle$ holds for all possible interpretations of ψ, ϕ and θ. Indeed it is simple to define such a unitary cloning map in **Rel**. (Note that the equations on scalars do not themselves imply the existance of a cloning map. Consider the free strongly compact closed category generated by the category **1** for a counterexample.)

To summarise: the no-cloning proof offered above shows that unitary cloning maps do not exist in the *free* theory of abstract quantum mechanics; therefore they may or may not exist in concrete interpretations of the free theory.

Chapter 7

The One-Way Quantum Computer

The one-way model of quantum computation was introduced by Raussendorf, Briegel and Browne [RB01, RBB02] to address concerns about the scalability of quantum information processing. A computation in the one-way model begins with a highly-entangled, multi-qubit state to which a sequence of single qubit measurements and local unitary corrections is applied. Since the choice of basis for each measurement may depend upon earlier measurements, an algorithm is specified by the pattern of measurements and corrections, and their dependence. In the course of the computation, the initial entangled state — called a cluster state — is destroyed by the measurements, hence the name "one-way".

This model contrasts strongly with the conventional idealised quantum computation which essentially consists of a large unitary operation, typically expressed as a series of quantum logic gates, applied to an array of qubits, followed by the simultaneous measurement of some set of those qubits. Despite being limited to single qubit unitaries, the one-way model is at least as powerful as the quantum circuit model, as it is possible to code any circuit as a pattern of measurements on a cluster state.

This chapter uses the one-way model as an extended example of the techniques of the preceding chapter applied in a concrete setting. We will define a polycategory modulo some equations and use circuits over this polycategory to represent computations in the one-way model. Due to their inherent parallelism, one-way computations are usually rather difficult to understand. However, by performing rewrites on the circuit representation we will prove the correctness of a number of such programs and reduce them to simple unitary maps. The one-way model is formalised in the *measurement calculus*, which we now define.

7.1 The Measurement Calculus

The *Measurement Calculus*, introduced by Danos, Kashefi and Panangaden [DKP07] is a formal rewriting system for the one-way model. It is compositional, both sequentially and in parallel, and is strongly normalising. Its terms have a denotational semantics in terms of completely positive maps. We review here the basic features of the calculus.

The syntax of the measurement calculus is based on the following *commands*:

- 2-qubit entanglement operator E_{ij};
- 1-qubit corrections X_i, Z_j;
- 1-qubit measurements M_i^α.

The indices i and j range over an array of qubits, on which the commands act. The entanglement operator E_{ij} is interpreted as the controlled-Z; the corrections are the usual Pauli X and Z operators. The command M_i^α is interpreted as a destructive projective measurement in the basis

$$|+_\alpha\rangle = \frac{1}{\sqrt{2}}(|0\rangle + e^{i\alpha}\,|0\rangle),$$
$$|-_\alpha\rangle = \frac{1}{\sqrt{2}}(|0\rangle - e^{i\alpha}\,|0\rangle).$$

To each measurement we associate a classical bit, called a *signal*; if the measurement of the ith qubit yields the $|+_\alpha\rangle$ state then $s_i = 0$, and conversely $s_i = 1$ if the state $|-_\alpha\rangle$ is produced.

Given signals s, t, define *dependent corrections* X_i^s and Z_j^t, by $X_i^1 = X_i$, $Z_j^1 = Z_j$ and $X_i^0 = Z_i^0 = I$; and *dependent measurements*

$${}^t[M_i^\alpha]^s = M_i^{(-1)^s\alpha + t\pi}.$$

Measurement commands are combined in *patterns*.

Definition 7.1. A *measurement pattern* consists of a finite set of qubits V with distinguished subsets $I, O \subseteq V$ of *inputs* and *outputs*, and a finite sequence of commands acting on V such that:

- no command depends on a signal from a measurement not yet performed;
- no command acts on a qubit already measured;
- qubits are measured if and only if they are not outputs.

These conditions ensure that a pattern represents a runnable sequential program. The input qubits I are prepared in some state $|\psi\rangle$, each of the non-inputs is initialised to the state $|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ and the commands are applied in sequence, leaving behind only the output qubits O.

After each measurement the pattern may proceed along two different computation paths depending on the outcome of the measurement. Such a path, coextensive with the command sequence, is called a *branch*. Each branch consists of a sequence of unitary maps and projection operators, hence it defines a CP-map. The execution of a pattern is a probabilistic choice over its branches. Since CP-maps are preserved under convex combination, the effect of running a pattern $\mathfrak{P}$ is to apply a CP-map $U_\mathfrak{P}$ to its input. In this case, we say that $\mathfrak{P}$ *implements* the map $U_\mathfrak{P}$, and two patterns are considered equivalent if they implement the same map.

Measurement patterns come equipped with a sound rewriting relation which is strongly normalising. This rewriting will not be treated here (see [DKP07] for details), however the normal forms enjoy an additional property which will be crucial to our development.

Proposition 7.2 (EMC normal form)**.** *For each measurement pattern $\mathfrak{P}$, there exists an equivalent pattern $\mathfrak{P}'$ such that $\mathfrak{P} \Rightarrow^* \mathfrak{P}'$, and in the command sequence of $\mathfrak{P}'$ all the entanglement operators precede all the measurements, which precede all the corrections.*

The measurement calculus is shown to be universal for unitary maps by representing a 2-element universal set of unitaries as patterns. Arbitrary unitaries can be built up compositionally and, by normalising the resulting patterns, time and space efficient implementations can be found. However it is rarely clear what a non-trivial measurement pattern computes. In the next section we approach this recognition problem by representing normal patterns in the circuit notation developed in the preceding chapter. Equational reasoning on the circuits allows the patterns to be transformed back into simple unitary representations without ancillary qubits.

7.2 Representing the Measurement Calculus

We will now introduce a polycategory $\mathcal{P}$ which we will use to encode the measurement calculus. Note that, as elsewhere in this thesis, we are not concerned with the branching aspect of pattern execution. Hence there is no need to represent any signals in the polycategory: we assume that the outcome of every measurement is known, and any dependent operations are evaluated accordingly. Given this simplification, the measurement command M^α may be replaced by a projection operator $|+_\alpha\rangle$ or $|-_\alpha\rangle$. Let $T_\alpha = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\alpha} \end{pmatrix}$; then

$$\begin{aligned} |+_\alpha\rangle &= |+\rangle\, T_{-\alpha}, \\ |-_\alpha\rangle &= |+\rangle\, T_{-\alpha+\pi} \end{aligned}$$

hence we will use only the projection $|+\rangle$, augmented by some T_α to represent the measurement commands, where $0 \leq \alpha < 2\pi$.

We make one further restriction: we consider only patterns whose entanglement graph is connected. In terms of syntax this means that there is at least one command of the form E_{ij} or E_{ji} for every qubit i. Any pattern which does not have this property may be expressed as a parallel composition of patterns which do, so this restriction serves only to simplify the analysis, without any loss of power.

Definition 7.3. Let $\mathcal{P}$ be the polycategory with a single object 1, and whose polyarrows are freely generated by polycomposition and symmetry from the following basic polyarrows:

$$\begin{array}{cc} |+\rangle : - \to 1 & T_\alpha : 1 \to 1 \\ \langle +| : 1 \to - & X : 1 \to 1 \\ E : 1,1 \to 1,1 & H : 1 \to 1 \end{array}$$

We write Z as a mnemonic for T_π. Since there is only one object there is only one identity arrow, which is also written 1; we write n as shorthand for $\underbrace{1,\ldots,1}_{n \text{ times}}$.

We interpret $\mathcal{P}$ in **FDHilb** via the symmetric monoidal functor $[\![\cdot]\!] : \mathcal{P} \to$ **FDHilb**, where $[\![1]\!] = Q$ and the comma is interpreted as the usual tensor product of Hilbert spaces. The functor assigns each polyarrow to a linear map such that

$$[\![|+\rangle]\!] = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix} : \mathbb{C} \longrightarrow Q$$

$$[\![\langle +|]\!] = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \end{pmatrix} : Q \longrightarrow \mathbb{C}$$

$$[\![E]\!] = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix} : Q \otimes Q \longrightarrow Q \otimes Q$$

$$[\![T_\alpha]\!] = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\alpha} \end{pmatrix} : Q \longrightarrow Q$$

$$[\![X]\!] = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} : Q \longrightarrow Q$$

$$[\![H]\!] = \frac{1}{2} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} : Q \longrightarrow Q$$

We will shortly form the category of circuits **Circ**($\mathcal{P}$); by the canonical embedding of $\mathcal{P}$ into **Circ**($\mathcal{P}$) we may use the graphical notation of circuits for arrows in the polycategory. The generators of $\mathcal{P}$ will be represented graphically as shown in Figure 7.1; polyarrows are connected acyclic graphs whose vertices are generators. As always, the diagrams should be read from top to bottom.

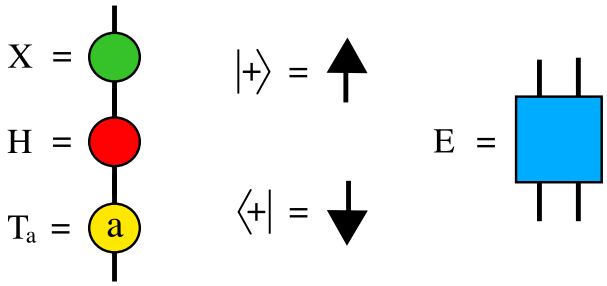


Figure 7.1: Generators for $\mathcal{P}$ in graphical form

Definition 7.4. Let $\mathfrak{P}$ be a pattern containing n measurements, and let $b \in \mathbb{B}^n$. We define the *branch* $(\mathfrak{P}, b)$ to be the pattern produced from $\mathfrak{P}$ by setting each signal s_i to the ith bit of b and evaluating all its dependent commands.

We now have sufficient material to construct a polyarrow to represent each branch. The construction proceeds in the obvious fashion, by substituting the generators of $\mathcal{P}$ for the corresponding measurement calculus commands. Sadly, there are two complications of a rather uninteresting, technical nature. Firstly, the polyarrow E requires its inputs to be adjacent, whereas the measurement command E_{ij} does not. Hence permutations must be introduced to shuffle the operands into the correct places. Secondly, care is required to ensure that a

valid polyarrow is defined. This will require some intermediate definitions, to which we now turn our attention.

Given an EMC-normal pattern $\mathfrak{P}$, let the *entangling sequence* $e(\mathfrak{P})$ be the subsequence of $\mathfrak{P}$'s commands containing exactly those commands which are entanglements. We use the notation $E^n_{i_n j_n}$ for the nth element of $e(\mathfrak{P})$ and let

$$e(\mathfrak{P})|_k = E^k_{i_k j_k} \cdots E^2_{i_2 j_2} E^1_{i_1 j_1} ,$$

so that $e(\mathfrak{P})|_k$ is the first k elements of $e(\mathfrak{P})$. Let

$$J_k(\mathfrak{P}) = \{i_1, j_1, \ldots, i_k, j_k\} ,$$

that is, $J_k(\mathfrak{P})$ is the set of qubit indices which occur in $e(\mathfrak{P})|_k$. Clearly $J_{k+1}(\mathfrak{P}) \subseteq J_k(\mathfrak{P})$, and if the length of $e(\mathfrak{P})$ is N then $J_N(\mathfrak{P}) = V$ since the entanglement graph is assumed to be connected. Since $J_k(\mathfrak{P}) \subset \mathbb{N}$, let $J_k(\mathfrak{P})$ be ordered as in $\mathbb{N}$.

Definition 7.5. A pattern $\mathfrak{P}$ is *topologically sorted* if, for all k, $i_{k+1} \in J_k(\mathfrak{P})$ or $j_{k+1} \in J_k(\mathfrak{P})$.

Lemma 7.6. *For every pattern $\mathfrak{P}$ there exists a pattern $\mathfrak{P}'$, equivalent to $\mathfrak{P}$, such that $\mathfrak{P}'$ is topologically sorted.*

Proof. Regardless of the values of i, j, k, l we have $E_{ij}E_{kl} = E_{kl}E_{ij}$ hence we may choose them in any order. Since the entanglement graph is connected, any topological sorting of the vertices will define a topologically sorted pattern. □

We will now concentrate upon patterns which are topologically sorted; thanks to Lemma 7.6 there is no loss of generality.

Given a topologically sorted pattern $\mathfrak{P}$, we now construct a polyarrow $f_n : |J_n(\mathfrak{P})| \to |J_n(\mathfrak{P})|$ for each entangling sequence $e(\mathfrak{P})|_n$ by recursion over n. The base case is trivial:

$$f_1 = E .$$

Suppose that f_n has already been constructed. Let $J_{n+1}(\mathfrak{P}) = \{\iota_1, \ldots, \iota_l\}$ so that the subscripts reflect the order, and let $i_{n+1} = \iota_k$ and $j_{n+1} = \iota_{k'}$; without loss of generality we assume $\iota_k < \iota_{k'}$.

Since $\mathfrak{P}$ is topologically sorted at least one of $\iota_k, \iota_{k'}$ belongs to $J_n(\mathfrak{P})$. Suppose both do; then $J_{n+1}(\mathfrak{P}) = J_n(\mathfrak{P})$. Define a permutation on $J_n(\mathfrak{P})$ by

$$\sigma = \begin{pmatrix} \iota_1 & \cdots & \iota_k & \iota_{k+1} & \cdots & \iota_{k'-1} & \iota_{k'} & \iota_{k'+1} & \cdots & \iota_l \\ \iota_1 & \cdots & \iota_k & \iota_{k'} & \iota_{k+1} & \cdots & \iota_{k'-1} & \iota_{k'+1} & \cdots & \iota_l \end{pmatrix}$$

Then let

$$f_{n+1} = (E \,{}_0{}^2{\circ}_{k-1}\, (f_n)^\sigma)^{\sigma^{-1}} .$$

Otherwise suppose that $\iota_k \in J_n(\mathfrak{P})$ but $\iota_{k'} \notin J_n(\mathfrak{P})$. Then define a permutation on $J_{n+1}(\mathfrak{P})$:

$$\tau = \begin{pmatrix} \iota_1 & \cdots & \iota_{k'-1} & \iota_{k'} & \iota_{k'+1} & \cdots & \iota_l \\ \iota_1 & \cdots & \iota_{k'-1} & \iota_{k'+1} & \cdots & \iota_l & \iota_{k'} \end{pmatrix}$$

Now define

$$f_{n+1} = (E \,{}_0{}^1{\circ}_{k-1}\, (f_n))^{\sigma^{-1}}_{\tau} .$$

The remaining case, when $\iota_{k'} \in J_n(\mathfrak{P})$ but $\iota_k \notin J_n(\mathfrak{P})$, is essentially the same as this one. All three are illustrated in Figure 7.2. Given a measurement pattern $\mathfrak{P}$, let $f_{e(\mathfrak{P})}$ be the polyarrow defined by $e(\mathfrak{P})$.

It is worth remarking that, while we have not defined the semantics of the measurement calculus, $[\![f_{e(\mathfrak{P})}]\!]$ necessarily defines the same unitary map as the entanglement sequence $e(\mathfrak{P})$. This is an immediate consequence of the fact that $[\![\cdot]\!]$ is symmetric monoidal.

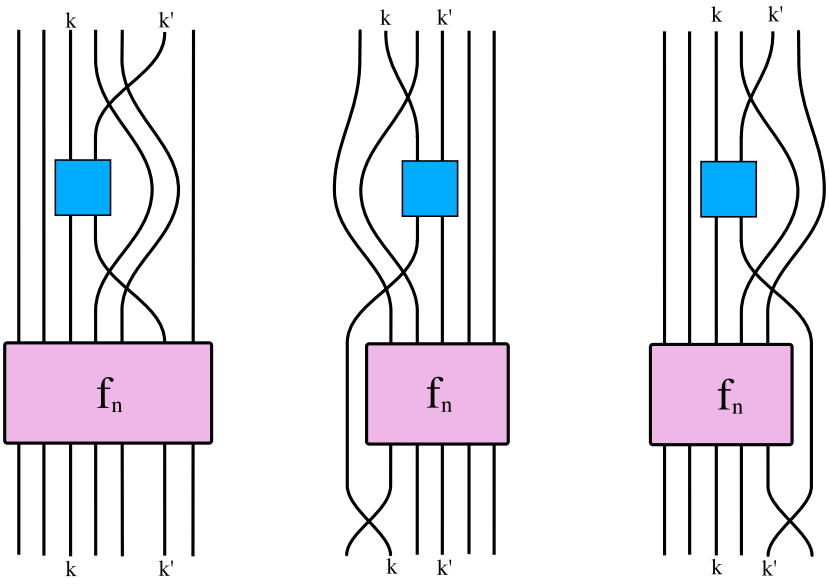


Figure 7.2: Defining a polyarrow from an entanglement sequence.

Definition 7.7. Let $(\mathfrak{P}, b)$ be a branch, where $\mathfrak{P}$ is a topologically sorted pattern in EMC-normal form. Let $i_1, \ldots i_m$ be the indices of its non-input qubits; let $j_1, \ldots, j_n$ be the indices of the qubits which are measured; and let $k_1, \ldots, k_p$ be the indices of the qubits which have corrections applied to them. We assume that measurements and non-inputs are listed in ascending order of index; and corrections are listed in the *reverse* order to that in which they occur in the pattern. We define a polyarrow

$$p(\mathfrak{P}, b) : |I| \longrightarrow |O|$$

by

$$\begin{aligned} p(\mathfrak{P}, b) = C^{(1)}{}_0 \overset{1}{\circ}_{k_1 - 1} \cdots C^{(p)}{}_0 \overset{1}{\circ}_{k_p - 1} \\ (\langle + |{}_0 \overset{1}{\circ}_0 T_{-\alpha_{j_1}}){}_0 \overset{1}{\circ}_{j_1 - 1} \cdots (\langle + |{}_0 \overset{1}{\circ}_0 T_{-\alpha_{j_n}}){}_0 \overset{1}{\circ}_{j_n - 1} \\ f_{e(\mathfrak{P})\, i_m - 1} \overset{1}{\circ}_0 |+\rangle \cdots {}_{i_1} \overset{1}{\circ}_0 |+\rangle \end{aligned}$$

where the measurement at the jth qubit is α_j and $C^{(k)}$ is either X or Z depending on the pattern,

This somewhat nightmarish composition looks more complicated than it is. The basic idea is simply to plug all the single qubit operations into $f_{e(\mathfrak{P})}$ in

the appropriate places. Recall that specification of a composition of polyarrows includes the number of objects at the left of the domain/codomain which are "skipped over". Composition with $\langle +|$ or $|+\rangle$ changes the arity of the polyarrow, and hence interferes with the next composition. The fiddly indexing ensures that arrows are composed "starting on the right" to avoid this problem.

Polyarrows are instrically 2-dimensional objects, so their representation as strings of algebraic symbols does not serve easy comprehension, as the above perhaps demonstrates. The results of Chapter 6 establish that the graphical language of circuits is a faithful representation of any compact polycategory, so we will prefer diagrams to algebraic expressions from here on.

By way of a simple example[1] the unitary matrix

$$J(\alpha) = \begin{pmatrix} 1 & e^{i\alpha} \\ 1 & -e^{i\alpha} \end{pmatrix}$$

is computed by the two qubit pattern $\mathfrak{J}(\alpha) = X_2^{s_1} M_1^{-\alpha} E_{12}$. Consider the branch $(\mathfrak{J}(\alpha), 0)$, i.e. the run where the outcome of the measurement is $|+_{-\alpha}\rangle$; the resulting polyarrow, $p(\mathfrak{J}(\alpha), 0)$, is shown in Figure 7.3 (a); $p(\mathfrak{J}(\alpha), 1)$ is is shown in Figure 7.3 (b).

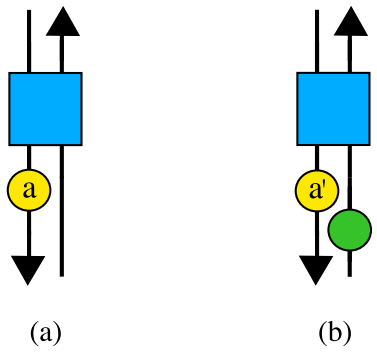


Figure 7.3: Example: the $\mathfrak{J}(a)$ pattern, with $a' = a + \pi$

It is easy to check that $J(\alpha) = [\![H \circ T_\alpha]\!]$. However, since $\mathcal{P}$ is freely generated, it lacks the necessary equations to prove that

$$H \circ T_\alpha = p(\mathfrak{J}(\alpha), b).$$

Hence we will consider a quotient of $\mathcal{P}$ by some carefully chosen equations. The equations chosen will hold in **FDHilb**, so the interpretation functor $[\![\cdot]\!]$ will factor uniquely through the quotient category to give a new interpretation $[\![\cdot]\!]_q$. In fact we will take the quotient of the free dagger polycategory on $\mathcal{P}$ so that the resulting class of circuits will be strongly compact closed.

Let $D\mathcal{P}$ denote the free dagger polycategory on $\mathcal{P}$; let $\mathcal{Q}$ denote the quotient category of $D\mathcal{P}$ by the following equations.

[1] Like all the examples of this chapter, this pattern is from [DKP07].

Adjoints

$$|+\rangle^\dagger = \langle +| \tag{7.1}$$
$$T_\alpha^\dagger = T_{-\alpha} \tag{7.2}$$
$$X^\dagger = X \tag{7.3}$$
$$H^\dagger = H \tag{7.4}$$
$$E^\dagger = E \tag{7.5}$$

Simplifications

$$T_\alpha \circ T_\beta = T_{\alpha+\beta} \tag{7.6}$$
$$T_0 = 1 \tag{7.7}$$
$$H \circ H = 1 \tag{7.8}$$
$$X \circ X = 1 \tag{7.9}$$
$$\langle +| \; {}_0\overset{1}{\circ}_1 \; E \; {}_0\overset{1}{\circ}_0 \; |+\rangle = H \tag{7.10}$$

Commuting Relations

$$E_\sigma^\sigma = E \tag{7.11}$$
$$E \; {}_0\overset{1}{\circ}_1 \; E = E \; {}_1\overset{1}{\circ}_0 \; E \tag{7.12}$$
$$E \; {}_0\overset{1}{\circ}_0 \; T_\alpha = T_\alpha \; {}_0\overset{1}{\circ}_0 \; E \tag{7.13}$$

The reader is invited to verify that, for every equation $F = G$ listed above, $[\![F]\!] = [\![G]\!]$ holds in **FDHilb**. The equations are clearer in graphical form, as shown in Figure 7.4. We can now prove, using the circuit representation, that

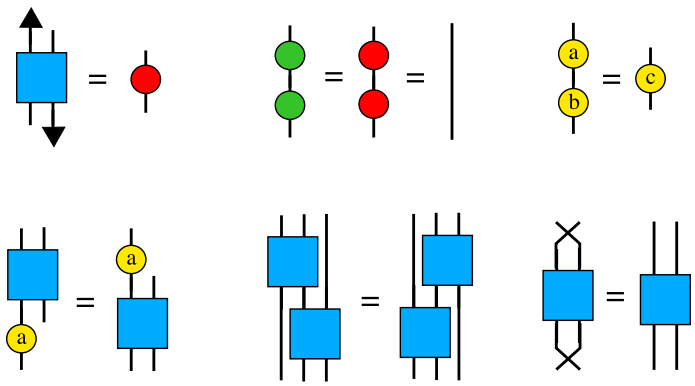
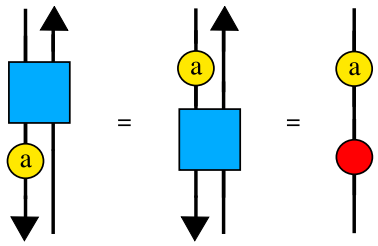


Figure 7.4: Equations for $\mathcal{P}$

the patterns for $J(\alpha)$ really compute the desired map.



Obviously $\mathcal{Q}$ is a more interesting than category $\mathcal{P}$; unfortunately, it is not strongly compact closed as our abstract quantum mechanics requires. Therefore we will work in $\mathbf{Circ}(\mathcal{Q})$, the free compact closed category on $\mathcal{Q}$. Note that as a consequence of Equations (7.1)–(7.5) every adjoint in $\mathcal{Q}$ can be expressed as an element in $\mathcal{P}$ hence we can neglect the dagger structure entirely.

We make one small but significant modification to the framework described in earlier chapters. In the context of the measurement calculus we consider only the concrete space of qubits Q and never its dual Q^*. To accommodate this we implicitly compose each unit map η_Q with the sesquilinear map which sends $z \mapsto \overline{z}$ for each complex number z. While this map is not linear, and hence not present in **FDHilb**, the resulting *pseudo-unit* $\overline{\eta_Q} : I \to Q \otimes Q$ is linear, and corresponds to the Bell state $|00\rangle + |11\rangle$ in **FDHilb**. The pseudo-counit $\overline{\epsilon_Q} : Q \otimes Q \to I$ is defined similarly.

Just as the unit and counit were used to define for each arrow $f : A \to B$ its dual $f^* : B^* \to A^*$, so $\overline{\eta_Q}$ and $\overline{\epsilon_Q}$ define the *transpose* $f^T : B \to A$. The notions of name $\ulcorner f \urcorner$ and coname $\llcorner f \lrcorner$ are similarly adapted. It is easy to see that $\overline{\eta_Q}, \overline{\epsilon_Q}$ and f^T satisfy the same equations as η_Q, ϵ_Q and f^*.

Given the comments above, we can define the transpose diagrammatically in $\mathbf{Circ}(\mathcal{Q})$. We note that each of our operations (except $|+\rangle$ and $\langle +|$) is invariant under transpose. Further, E is invariant under partial transpose. These equations are shown in Figure 7.5.

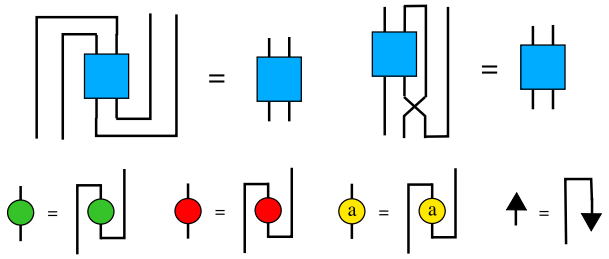
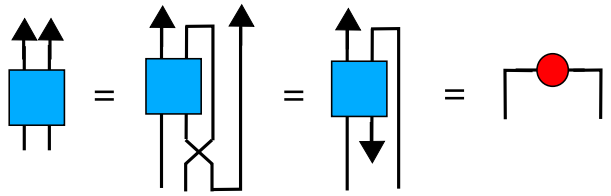


Figure 7.5: Equations for transpose invariance in $\mathbf{Circ}(\mathcal{Q})$

Lemma 7.8. *In* $\mathbf{Circ}(\mathcal{Q})$ *we have*

$$E\,|{+}{+}\rangle = \ulcorner H \urcorner$$
$$\langle {+}{+}|\,E = \llcorner H \lrcorner$$

Proof. See the diagram below.



The equalities are due to: the invariance of E under partial transpose; topological simplification; and, Equation (7.10). The other claim is dual. □

7.3 Examples

The paper [DKP07] provides several examples of measurement calculus patterns to compute various unitary maps. In the remainder of the chapter we will use the circuit representation to prove that these patterns are indeed correct. Starting with the circuit representation of a pattern, we rewrite it according to the equations above until a manifestly unitary circuit is all that remains.

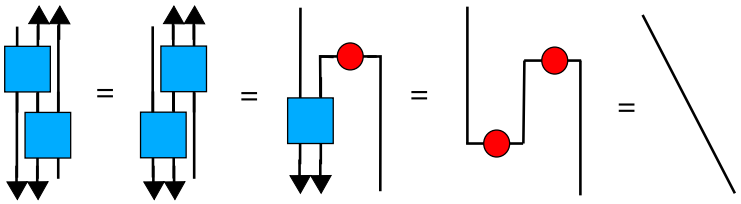
Remark. In each case we will consider only a single branch, the branch where all signals are zero. A proof that the pattern meets its specification would normally require every branch to be treated. However the examples we have chosen enjoy the property of *flow* which implies that all their branches coincide, and hence the proofs given here are complete. See [DK05] for details.

7.3.1 Teleportation

The pattern for teleportation map is

$$X_3^{s_2} Z_3^{s_1} M_2^0 M_1^0 E_{23} E_{12} \ .$$

This pattern transfers the input state on qubit 1 to qubit 3; if we don't distinguish between the qubits then the underlying map of the teleportation protocol is the identity. Translating into $\mathbf{Circ}(\mathcal{Q})$, we have the circuit shown at the left, below.



7.3.2 One qubit unitary

Any unitary map is equivalent to a sequence of three rotations $R_x(\gamma)R_z(\beta)R_x(\alpha)$, which in turn may be factorised [DKP04] as

$$R(\alpha, \beta, \gamma) = J(0)J(\alpha)J(\beta)J(\gamma).$$

which results in the standard pattern:

$$X_5^{s_2+s_4} Z_5^{s_1+s_3} M_4^0 [M_3^{-\alpha}]^{s_2} [M_2^{-\beta}]^{s_1} M_1^{-\gamma} E_{45} E_{34} E_{23} E_{12} \ .$$

Since all the signals are set to zero, this results in the circuit shown at the top left of the diagram below. By the indicated sequence of rewrites, this circuit can be reduced to the composition of one qubit gates shown in the bottom right. Note that $J(\alpha) = HT_\alpha$, hence

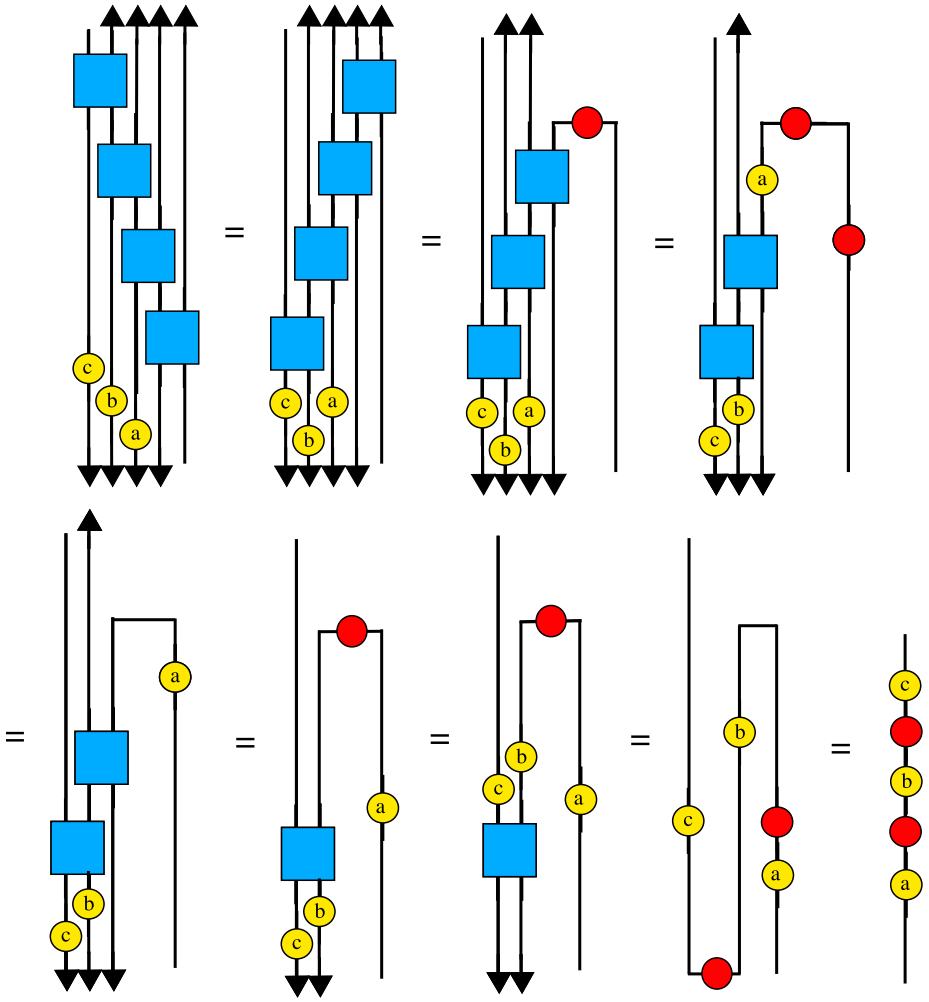


Figure 7.6: The pattern for an arbitrary unitary, $R(a, b, c)$.

$$J(0)J(\alpha)J(\beta)J(\gamma) = HT_0HT_\alpha HT_\beta HT_\gamma = T_\alpha HT_\beta HT_\gamma$$

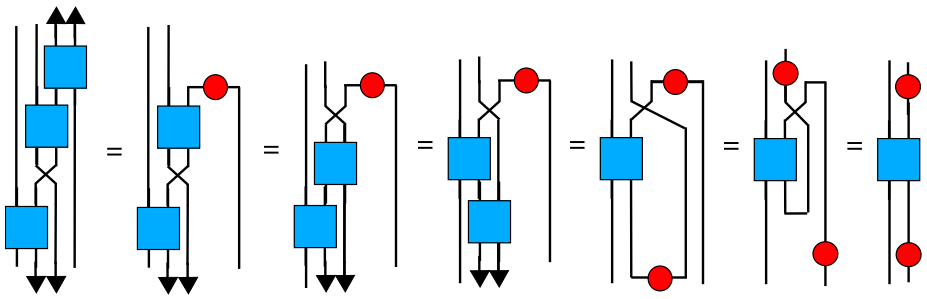
which is the map depicted in Figure 7.6. Therefore the pattern implements the unitary as required.

7.3.3 Controlled-NOT

The pattern

$$X_4^{s_3} Z_4^{s_2} Z_1^{s_2} M_3^0 M_2^0 E_{13} E_{23} E_{34}$$

implements the controlled-not gate. This translation into $\mathbf{Circ}(\mathcal{Q})$ yields the following rewrite sequence:



whose correctness is easily verified.

7.3.4 Controlled-U

The final example is the controlled unitary gate, for an arbitrary one-qubit unitary U. Again, borrowing from [DKP04], we construct the CU map as:

$$CU = T_1^{\alpha'} T_2^{\beta+\pi} H_2 T_2^{-\frac{\gamma}{2}} H_2 T_2^{-\frac{\pi}{2}} H_2 CZ_{12} \\ H_2 T_2^{\frac{\pi}{2}} H_2 T_2^{\frac{\gamma}{2}} H_2 T_2^{\frac{-\pi-\delta-\beta}{2}} H_2 CZ_{12} H_2 T_2^{\frac{-\beta+\delta-\pi}{2}}. \quad (7.14)$$

with $\alpha' = \alpha + \frac{\beta+\gamma+\delta}{2}$. The required pattern has fourteen qubits, indexed by lower case letters $a, b, c, \ldots, k$ and upper case letters A, B and C. The normalised pattern $\mathfrak{CU}$ is shown below.

$$Z_k^{s_i+s_g+s_e+s_c+s_a} X_k^{s_j+s_h+s_f+s_d+s_b} X_C^{s_B} Z_C^{s_A+s_e+s_c} \\ M_B^0 M_A^{-\alpha'} M_j^0 [M_i^{\beta-\pi}]^{s_h+s_f+s_d+s_b} [M_h^{-\frac{\gamma}{2}}]^{s_g+s_e+s_c+s_a} [M_g^{\frac{\pi}{2}}]^{s_f+s_d+s_b} \\ M_f^0 [M_e^{-\frac{\pi}{2}}]^{s_d+s_b} [M_d^{\frac{\gamma}{2}}]^{s_c+s_a} [M_c^{\frac{\pi-\delta-\beta}{2}}]^{s_b} M_b^0 M_a^{\frac{-\beta+\delta+\pi}{2}} \\ E_{BC} E_{AB} E_{jk} E_{ij} E_{hi} E_{gh} E_{fg} E_{Af} E_{ef} E_{de} E_{cd} E_{bc} E_{ab} E_{Ab}$$

The pattern is simplified by evaluating all the signals. Let $b = 0, 0, 0, 0, \ldots, 0$; then, the run $(\mathfrak{CU}, b)$ has the form shown below.

$$M_B^0 M_A^{-\alpha'} M_j^0 M_i^{\beta-\pi} M_h^{-\frac{\gamma}{2}} M_g^{\frac{\pi}{2}} M_f^0 M_e^{-\frac{\pi}{2}} M_d^{\frac{\gamma}{2}} M_c^{\frac{\pi-\delta-\beta}{2}} M_b^0 M_a^{\frac{-\beta+\delta+\pi}{2}} \\ E_{BC} E_{AB} E_{jk} E_{ij} E_{hi} E_{gh} E_{fg} E_{Af} E_{ef} E_{de} E_{cd} E_{bc} E_{ab} E_{Ab}.$$

The proof that this pattern above is equivalent to control-U gate as shown in Figures 7.7 and 7.8. Note that due to space restrictions the T_α elements are labelled with the index of their qubit rather than the angle found in the above expression; the angles can be read from the pattern, above.

7.4 Remark

The equations we have chosen above suffice to prove the correctness of the examples we present in this chapter. However it should be clear that these equations will not suffice to prove in $\mathcal{Q}$ every equation which holds in the image of $[\![\cdot]\!]$. A simple example is $HXH = Z$; the equation holds in **FDHilb**, but it cannot be shown in the abstract setting of $\mathcal{Q}$. A minimal set of equations which suffice to prove every identity which holds in the image $\mathcal{Q}$ has yet to be found.

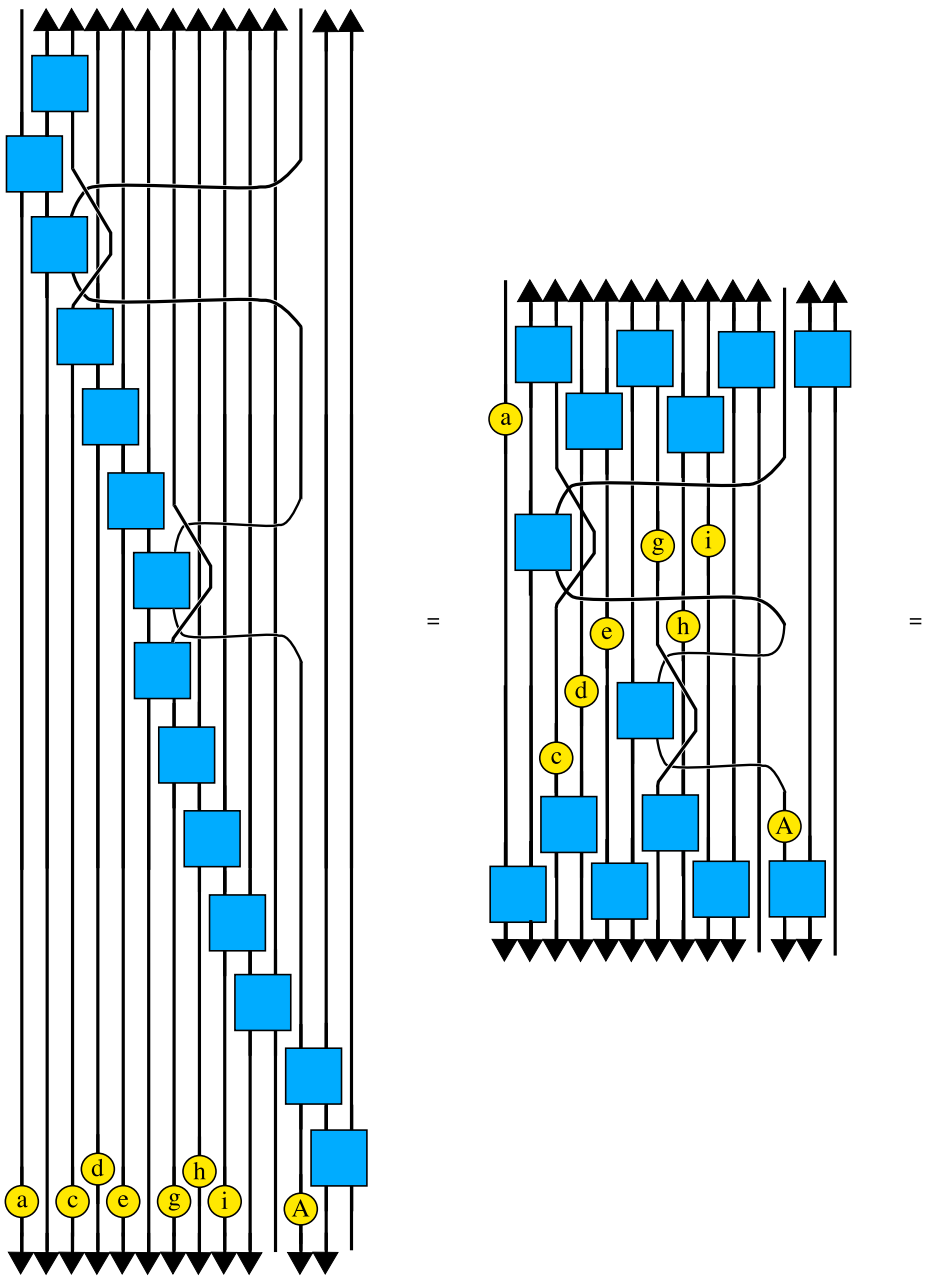


Figure 7.7: The controlled-U gate, part 1.

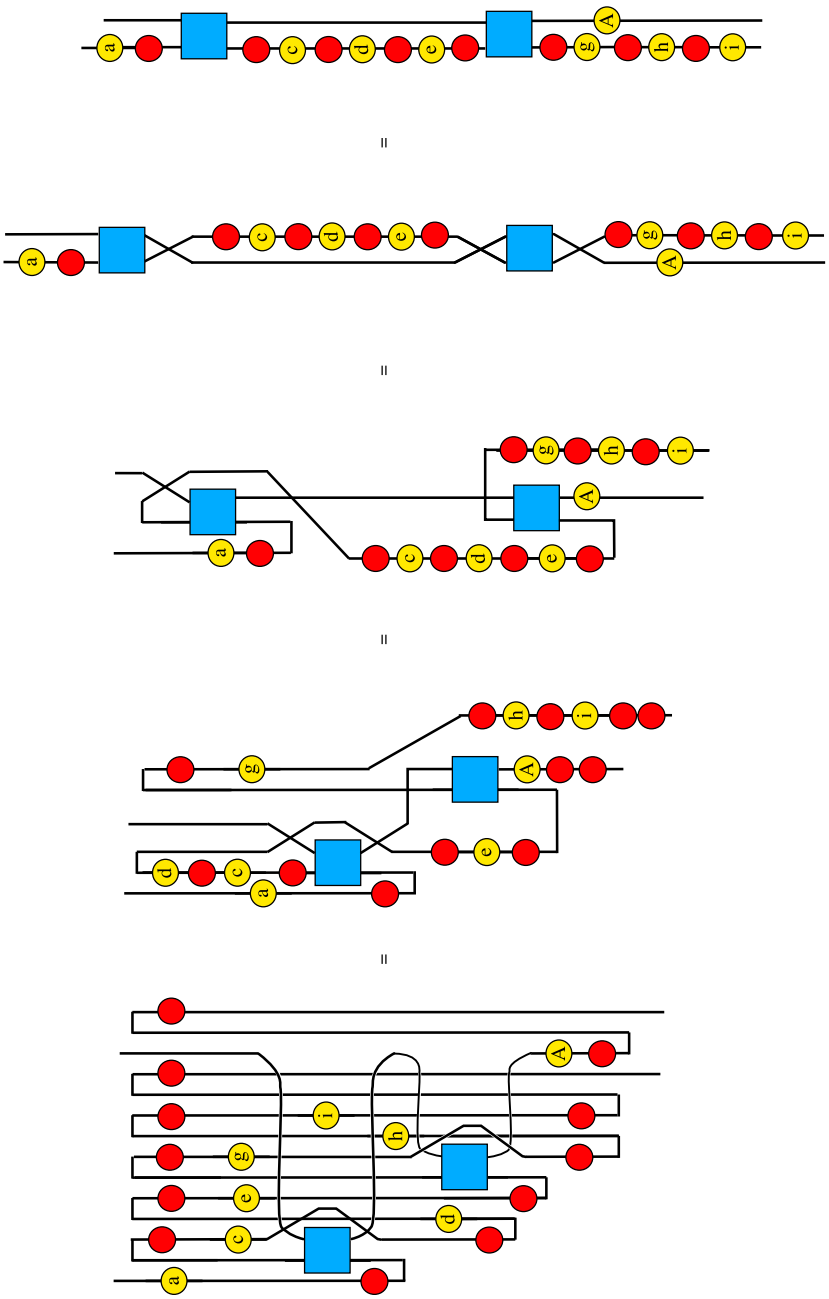


Figure 7.8: The controlled-U gate, part 2.

Chapter 8

Further Work

The primary objective of the this work was to describe the type theoretical basis of quantum computation. The preceding chapters have constructed suitable formal models based on the formulation of quantum mechanics in the framework of compact closed categories. This represents a significant contribution towards the construction of a quantum programming language based explicitly on the behaviour of quantum systems. This work suggests several future directions for further research.

Formal Quantum Mechanics

The framework elaborated here captures only the *multiplicative* aspect of quantum mechanics. To give a complete account of quantum mechanics the framework must be extended to include the probabilistic branching of quantum measurement; for use as a programming framework classical information and control, including classical probabilities, should also be incorporated.

In the original paper [AC04], these features are represented using a *biproduct*, a degenerate version of linear logic's additive connectives. The simple proof-net formalism of Chapter 4 is extended with biproducts in [AD06], and the techniques there are readily adapted to the generalised proof-nets of Chapter 6. The cited work divides the proof-net into a number of *slices* (cf. [Gir87a]) to represent the additive structure; a more practical approach uses local additive boxes, so that proof-nets may be composed without requiring any knowledge beside their types.

The biproduct is not the only candidate structure for this role. Coecke and Pavlovic [CP07] introduce *classical objects* and use this notion to express the branching behaviour of measurements in a coalgebraic framework. Selinger [Sel06] uses the idea of splitting idempotents to incorporate classical types into a purely multiplicative framework. Both of these formalisms have the benefit of obviating the connective manipulation required by the separation of multiplicative and additive aspects, however the representation of control in these approaches is less obvious.

Intimately connected with probabilistic behaviour in quantum mechanics is the notion of mixed state. The constructions presented here are based on compact closed structure only, and hence are readily applicable to both pure and mixed state quantum computation, although we have focused on the for-

mer here. In other work Selinger [Sel05] and Coecke [Coe05] have considered the construction of a category of mixed states from a category of pure states. Since any practical quantum computation device is likely to deal with mixed states it seems a worthwhile exercise to investigate these constructions in proof-theoretical terms.

Entanglement and Computation

Current proposals for quantum programming languages do not take entanglement seriously. The structure inside the quantum register is largely neglected. In Chapter 5 we saw how the type system of **MLL** can assign types to **mCQL** terms such that type describes the entanglement within the corresponding quantum state. Of course, **mCQL** has limited expressive power with respect to entangled states, so the generalised proof-nets of Chapter 6 are a more appropriate tool to give the informatic flows within the quantum realm a more complete treatment. If we restrict attention to one-sided nets — that is, to nets denoting quantum states rather than maps between them — the same intuition for assigning **MLL** types works well. One can construct an **MLL** type for a generalised proof-net by replacing some of the tensor links with par links. However given that the graph structure of a generalised proof-net is arbitrarily complicated, it remains to find a correctness criterion similar to the Danos-Regnier criterion [DR89].

Of course a programming system must describe functions as well as states. An appealing idea is to assign a type to each function, denoting whether it creates or destroys entanglement; however it is easy to find problematic examples. The controlled-Z gate can map a separable state to a maximally entangled pair, but only for certain inputs; other separable states remain separable. Worse, this example is an involution, and hence not even monotonic with respect to entanglement. Sticking with the **MLL** typing discipline we are led to interpret $\otimes$ as "surely separable" which is a subtype of $\parr$, understood as as "maybe entangled", as seen in [Dun04, BIP03, Per05]. I am pessimistic as to whether this approach will lead to more informative judgements than "maybe-entangled" as both input and output for any non-trivial term.

However the treatment of the measurement calculus developed in Chapter 7 suggests one avenue to progress. There, the circuit formalism was augmented with equations reflecting properties of the Hilbert space model. The set of equations used for rewriting is not complete, however the maps T_α, H, X and CZ do generate all the unitaries in **FDHilb**. The discovery of a set of equations $\mathcal{E}$ on these elements which made the interpretation functor $\mathbf{Circ}(\mathcal{Q})/\mathcal{E} \to \mathbf{FDHilb}$ faithful would be a major step. By orienting these equations as rewrite rules, a formal system for reducing measurement patterns to their specification will be defined; of course, small details like confluence and termination must be taken care of as well to produce a truly effective system.

However, it is not necessary to go quite so far. The key property of the the circuit representation of a quantum state is connectedness. Any equation which rewrites a connected fragment to a disconnected one will yield a better approximation to the true entanglement in the underlying quantum state. Hence it is not necessary to derive every equation which holds in **FDHilb**, but merely those which hold between graphs of differing topologies. Whether this is actually easier remains to be seen, but one recent result which seems relevant is [AP05].

It is worth remarking that our proof that the pattern $\mathfrak{CU}$ computes the controlled-U map did so by reducing the pattern essentially to a sequence of teleportation and "pushing" the phase and Hadamard maps through this network. This suggests that it may be possible to use this technique to prove the converse to the flow theorem: that if a pattern computes a unitary map then it must contain a "flow" from its inputs to its outputs.

The preceding discussion assumes that the dichotomy between entangled and separable states will be sufficient for a practical type system. A more subtle discrimination may be desirable for some applications. As we noted in Chapter 5, in the abstract setting maximal entanglement in a state encodes unitarity in the corresponding map. In the generalised setting, even if all the generators are unitary there are non-unitary maps, namely unitaries with part of their domain traced out. Hence there are states which are entangled, but less than maximally so. This topic seems ripe for further investigation. The divergence from unitarity of a circuit is related to how many traces have been taken over it: that is, how many loops are present. This is an essentially topological feature; it seems possible that the tools of algebraic topology may provide what is needed for an abstract classification of entanglement. More concretely, it will be necessary to relate the abstract entanglements found in the free model, to the concrete measures of entanglement found in the quantum information literature.

Quantum Processes

We have generally assumed strictly monoidal categories, though the proof-net formalism could easily adapt to a non-strict setting, perhaps with an increase in elegance. Quantum protocols rely on a notion of *agent* which we could view as a *bracketing* of the global state space. Under such an interpretation the associativity isomorphisms correspond to the redistribution of quantum resources among the agents. For example, in a non-strict setting, the triangle law of compact closed categories,

$$\begin{array}{ccccc} A & \xrightarrow{\rho} & A \otimes I & \xrightarrow{1_A \otimes \eta_A} & A \otimes (A^* \otimes A) \\ \Big\downarrow{\scriptstyle 1_A} & & & & \Big\downarrow{\scriptstyle \alpha} \\ A & \xleftarrow[\lambda^{-1}]{} & I \otimes A & \xleftarrow[\epsilon \otimes 1_A]{} & (A \otimes A^*) \otimes A, \end{array}$$

can be interpreted as a more explicit variation of the teleportation protocol where Bob, on the right hand side of the tensor, creates the entangled pair and sends one qubit to Alice. The transmission of this qubit is encoded by the associativity map α.

In this way, a representation of a quantum process naturally splits into two parts: the entanglement structure, which is captured by the process's representation as a circuit or proof-net in the sense of Chapter 6; and its decomposition into systems and subsystems. Notice that such a decomposition is necessary to any discussion of entanglement, indeed the definition of entanglement requires it. In Chapter 5 we took a somewhat ad hoc approach and used the parse tree of the type as the state's decomposition. In a more realistic setting, where agents

pass quantum resources between themselves, the distribution of the state would not be a static in this way, but rather a dynamic part of the system's behaviour.

A more promising candidate for a serious approach to this problem is Milner's theory of *bigraphs* [MJ04, Mil01]. Bigraphical reactive systems are a very general class of interaction calculi which explicitly represent locality of agents as well as their interconnections. The integration of the logical characterisation of quantum processes carried out in this thesis and the more general setting of bigraphs appears a profitable approach to a quantum process algebra, or a quantum programming language with explicit concurrency, not least because bigraphs can be formalised in a rich categorical setting based on 2-categories [SS02, SS04].

Algebraic Generalisations

Moving away from concrete quantum computation, recall that our work has been carried out in the setting of symmetric monoidal categories. Relaxing this condition to consider planar, braided or ribbon categories may yield interesting structures and help make connections with areas as diverse as quantum groups and cyclic linear logic. In particular there is a striking formal resemblance between the *spin networks* used in some approaches to quantum gravity [Bae98], and the representation of arrows in $\mathbf{Circ}(\mathcal{A})$. This connection merits fuller investigation. While spin networks live in an n-categorical framework, there is a strong connection between polycategories and bicategories [Lam05, CKS00]. It is perhaps worth the effort to recast the theory in this setting; the notion of computads [Bat02, Mak05] may be useful for the free case.

Another important connection to be developed is that between the entanglement of a state and the complexity of program used to produce it. Two results of Richard Jozsa are relevant here. Firstly, in [JL03], Jozsa and Linden establish that an exponential speedup is only achieved in programs which create an asymptotically unbounded amount of entanglement amongst its subsystems. A more recent result [Joz06] states that the computational cost of simulating a quantum circuit is exponential in the number of swaps in the circuit. From the abstract point of view there is no essential difference between a quantum state and the process which produces it. Hence I hypothesise that the deviation of a graph from planarity – the abstract counterpart to the swaps – will give rise to a natural measure of complexity. Some related work by Metayer [Mét94, Mét01] and Gaubert [Gau04] makes connections between the topological genus implicitly defined by a proof in multiplicative linear logic and the number of exchanges contained therein. Since the logic of compact closed categories is a close relative of multiplicative linear logic it is likely that this work will transfer easily to the quantum setting.

Bibliography

[Aar05] S. Aaronson. Quantum computing, postselection, and probabilistic polynomial-time. *Proceedings of the Royal Society A*, 461(2063):3473–3482, 2005, quant-ph/0412187.

[ABP99] S. Abramsky, R. Blute, and P. Panangaden. Nuclear and trace ideals in tensored *-categories. *Journal of Pure and Applied Algebra*, 143:3–47, 1999.

[Abr93] S. Abramsky. Computational interpretations of linear logic. *Theoretical Computer Science*, 111(1-2):3–57, 1993.

[Abr05] S. Abramsky. Abstract scalars, loops, and free traced and strongly compact closed categories. In J. Fiadeiro, editor, *Proceedings of CALCO 2005*, volume 3629 of *Springer Lecture Notes in Computer Science*, pages 1–31, 2005.

[AC04] S. Abramsky and B. Coecke. A categorical semantics of quantum protocols. In *Proceedings of the 19th Annual IEEE Symposium on Logic in Computer Science: LICS 2004*, pages 415–425. IEEE Computer Society, 2004.

[AC05] S. Abramsky and B. Coecke. Abstract physical traces. *Theory and Applications of Categories*, 14(6):111–124, 2005.

[AD06] S. Abramsky and R. Duncan. A categorical quantum logic. *Mathematical Structures in Computer Science*, 16:469–489, 2006, arXiv:quant-ph/0512114. Special Issue for the Proceedings of QPL 2004.

[AG05] T. Altenkirch and J. Grattage. A functional quantum programming language. In *Proceedings of Logic in Computer Science*, 2005.

[AGN95] S. Abramsky, S. Gay, and R. Nagarajan. Specification structures and propositions-as-types for concurrency. In F. Moller and G. Birtwistle, editors, *Logics for Concurrency: Structure vs Automata — Proceedings of the Banff Higher Order Workshop*. Springer-Verlag Lecture Notes in Computer Science, 1995.

[AGN96] S. Abramsky, S. Gay, and R. Nagarajan. Interaction categories and the foundations of typed concurrent programming. In M Broy, editor, *Proceedings of the 1994 Marktoberdorf Summer School on Deductive Program Design*, pages 35–113. Springer-Verlag, 1996.

[AHS02] S. Abramsky, E. Haghverdi, and P. Scott. Geometry of interaction and linear combinatory algebras. *Mathematical Structures in Computer Science*, 12:625–665, 2002.

[AJ94] S. Abramsky and R. Jagadeesan. New foundations for the geometry of interaction. *Information and Computation*, 111(1):53–119, 1994. Conference version appeared in LiCS '92.

[AK] S. Aaronson and G. Kuperberg. The complexity zoo. `http://qwiki.caltech.edu/wiki/Complexity_Zoo`.

[AP05] K.M.R. Audenaert and M. B. Plenio. Entanglement on mixed stabiliser states–i: Normal forms and reduction procedures. *New J. Phys*, 7, 2005, quant-ph/0505036.

[Bae98] J. Baez. Spin foam models. *Class. Quantum Grav*, 15:1827–1858, 1998, gr-qc/9709052.

[Bar79] M. Barr. $*$-*Autonomous Categories*, volume 752 of *Lecture Notes in Mathematics*. SpringerVerlag, 1979.

[Bar84] H. Barendregt. *The Lambda Calculus: Its Syntax and Semantics*. North-Holland Publishing, 1984.

[Bar91] M. Barr. *-autonomous categories and linear logic. *Mathematical Structures in Computer Science*, 1:159–178, 1991.

[Bat02] M.A. Batanin. Computads and slices of operads, 2002, math.CT/0209035.

[BBC+93] C. Bennett, G. Brassard, C. Crepeau, R. Jozsa, A. Peres, and W. Wootters. Teleporting an unknown quantum state via dual classical and EPR channels. *Phys. Rev. Lett.*, pages 1895–1899, 1993.

[BBPS96] C. H. Bennett, H. J. Bernstein, S. Popescu, and B. Schumacher. Concentrating partial entanglement by local operations. *Phys. Rev. A*, 53(4):2046–2052, 1996.

[BCST96] R.F. Blute, J.R.B Cockett, R.A.G. Seely, and T.H. Trimble. Natural deduction and coherence for weakly distributive categories. *Journal of Pure and Applied Algebra*, 113:229–296, 1996.

[BD95] J. Baez and J. Dolan. Higher-dimensional algebra and topological quantum field theory. *J.Math.Phys*, 36:6073–6105, 1995, q-alg/9503002.

[BIP03] R. F. Blute, I. T. Ivanov, and P. Panangaden. Discrete quantum causal dynamics. *Int. J. Theor. Phys.*, 42:2025–2041, 2003.

[Blu93] R. Blute. Linear logic, coherence and dinaturality. *Theoretical Computer Science*, 115:3–41, 1993.

[BS94] G. Bellin and P. J. Scott. On the π-calculus and linear logic. *Theoretical Computer Science*, 1994.

[BS04] A Baltag and S. Smets. The logic of quantum programs. In Peter Selinger, editor, *Proceedings of the 2nd International Workshop on Quantum Programming Languages*, number 33 in TUCS General Publication. Turku Centre for Computer Science, July 2004.

[BvN36] G. Birkhoff and J. von Neumann. The logic of quantum mechanics. *Annals of Mathematics*, 37(4):823–843, October 1936.

[CF58] H. B. Curry and R. Feys. *Combinatory Logic I.* North-Holland Publishing, 1958.

[Chu40] A. Church. A formulation of the simple theory of types. *Journal of Symbolic Logic*, 5, 1940.

[CKS00] J.R.B Cockett, J. Koslowski, and R.A.G. Seely. Introduction to linear bicategories. *Mathematical Structures in Computer Science*, 10(2):165–203, 2000.

[Cle00] R. Cleve. An introduction to quantum complexity theory. In C. Macchiavello, G .M. Palma, and A. Zeilinger, editors, *Collected Papers on Quantum Computation and Quantum Information Theory*. World Scientific, 2000, arXiv:quant-ph/9906111.

[Coe05] B. Coecke. De-linearizing linearity: Projective quantum axiomatics from strong compact closure. In *Proceedings of the 3rd International Workshop on Quantum Programming Languages*, 2005, arXiv:quant-ph/0506134.

[CP06] B. Coecke and E. O. Paquette. POVMs and Naimark's theorem without sums. In *Proceedings of the 4th International Workshop on Quantum Programming Languages*, 2006.

[CP07] B. Coecke and D. Pavlovic. Quantum measurements without sums. In G. Chen, L. H. Kauffman, and Jr Lomonaco, S.J., editors, *The Mathematics of Quantum Computation and Technology*, CRC Applied Mathematics & Nonlinear Science. Taylor and Francis, 2007, quant-ph/0608035.

[CS97] R. Cockett and R.A.G. Seely. Weakly distributive categories. *Journal of Pure and Applied Algebra*, 113(2):229–296, 1997.

[DK05] V. Danos and E. Kashefi. Determinism in the one-way model. In *ERATO conference on Quantum Information Science 2005*, 2005, arXiv.org:quant-ph/0506062.

[DKP04] V. Danos, E. Kashefi, and P. Panangaden. Robust and parsimonious realisations of unitaries in the one-way model. *Phys. Rev. A*, 72(6):060101, December 2004, quant-ph/0411071.

[DKP07] V. Danos, E. Kashefi, and P. Panangaden. The measurement calculus. *Journal of ACM*, 2007, arXiv:quant-ph/0412135. (to appear).

[DP05] E. D'Hondt and P. Panangaden. The computational power of the W and GHZ states. *Journ. Quantum Inf. and Comp*, 6(2):173–183, 2005, quant-ph/0412177.

[DR89] V. Danos and L. Regnier. The structure of multiplicatives. *Arch. Math. Logic*, 28(3):181–203, 1989.

[Dun04] R. Duncan. Believe it or not, bell states are a model of multiplicative linear logic. Technical Report PRG-RR-04-18, Oxford University Computing Laboratory, 2004.

[Eke91a] A. K. Ekert. *Correlations in Quantum Optics*. PhD thesis, Oxford University, 1991.

[Eke91b] A. K. Ekert. Quantum cryptography based on Bell's theorem. *Phys. Rev. Lett.*, 67(6):1769–1782, 1991.

[Fey82] R. P. Feynman. Simulating physics with computers. *Int. J. Theor. Phys.*, 21(6/7), 1982.

[FGM01] M. Fitzi, N. Gisin, and U. Maurer. Quantum solution to the byzantine agreement problem. *Phys. Rev. Lett.*, 87(21), 2001.

[Fuc02] C. Fuchs. Quantum mechanics as quantum information (and only a little more), 2002, quant-ph/0205039.

[Gau04] C Gaubert. Two-dimensional proof-structures and the exchange rule. *Mathematical Structures in Computer Science*, 14:73–96, 2004.

[GC99] D. Gottesman and I. L. Chuang. Quantum teleportation is a universal computational primitive. *Nature*, 402:390, 1999.

[Gen35] G. Gentzen. Untersuchungen über das schliessen. In M.E Szabo, editor, *The Collected Papers of Gerhard Gentzen*. North-Holland Publishing, 1935.

[Gir87a] J.-Y. Girard. Linear logic. *Theoretical Computer Science*, 50(1), 1987.

[Gir87b] J.-Y. Girard. Multiplicatives. In G. Lolli, editor, *Logic and Computer Science: New Trends and Applcations*, pages 11–37. Rendiconti del Seminario Mathematico dell'Universita e Politecnico di Torino, 1987.

[Gir95] J.-Y. Girard. Linear logic: its syntax and semantics. In Jean-Yves Girard, Yves Lafont, and Laurent Regnier, editors, *Advances in Linear Logic*, volume 222 of *London Mathematical Society Lecture Notes*. Cambridge University Press, 1995.

[Gir96] J.-Y. Girard. Proof-nets: the parallel syntax for proof theory. In Marcel Dekker, editor, *Logic and Algebra*. 1996.

[Gis91] N. Gisin. Bell's inequality holds for all non-product states. *Phys. Lett. A*, 154(5,6):201–202, April 1991.

[GLT89] J.-Y. Girard, Y. Lafont, and P. Taylor. *Proofs and Types*. Cambridge University Press, 1989.

[Gro96] L. K. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*, pages 212–219. ACM, 1996.

[GSS91] J.-Y. Girard, A. Scedrov, and P. J. Scott. Normal forms and cut-free proofs as natural transformations. In Y. N. Moschovakis, editor, *Proceedings Workshop Logic from Computer Science, Berkeley, CA, USA, 13–17 Nov 1989*, volume 21, pages 217–241. Springer-Verlag, New York, 1991.

[Hat02] A. Hatcher. *Algebraic Topology*. Cambridge University Press, 2002.

[Hin97] R.J. Hindley. *Basic Simple Type Theory*, volume 42 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 1997.

[Hin04] P. Hines. Can the von Neumann architecture be made quantum? manuscript, 2004.

[Hof99] M. Hoffman. Linear types and non-size increasing polynomial time computation. In *Proceedings of the 14th Symposium on Logic in Computer Science*. IEEE, 1999.

[HS03] M. Hyland and A. Schalk. Gluing and orthogonality for models of linear logic. *Theoretical Computer Science*, 294:183–231, 2003.

[HvG03] D. Hughes and R. van Glabbeek. Proof nets for unit-free multiplicative-additive linear logic. In *Proc. Logic in Computer Science 2003*. IEEE, 2003.

[IB00] C. J. Isham and J. Butterfield. Some possible roles for topos theory in quantum theory and quantum gravity. *Foundations of Physics*, 30:1707 – 1735, 2000.

[Ish95] C. J. Isham. *Lectures on Quantum Theory: Mathematical and Structural Foundations*. Imperial College Press, 1995.

[JL03] R. Jozsa and N. Linden. On the role of entanglement in quantum computational speed-up. *Proceedings of the Royal Society A*, 459:2011–2032, June 2003, arxiv:quant-ph/020143.

[Joy77] A Joyal. Remarques sur la théories des jeux à deux personnes. *Gazette des sciences mathématiques du Québec*, 1(4), 1977.

[Joz06] R. Jozsa. On the simulation of quantum circuits, 2006, quant-ph/0603163.

[JS91] A. Joyal and R. Street. The geometry of tensor categories i. *Advances in Mathematics*, 88:55–113, 1991.

[JS93] A. Joyal and R. Street. Braided tensor categories. *Advances in Mathematics*, 102:20–78, 1993.

[JSV96] A. Joyal, R. Street, and D. Verity. Traced monoidal categories. *Math. Proc. Camb. Phil. Soc.*, 119:447–468, 1996.

[Kel72a] G.M. Kelly. An abstract approach to coherence. volume 281 of *Lecture Notes in Mathematics*, pages 106–147. Springer, 1972.

[Kel72b] G.M. Kelly. Many-variable functorial calculus I. volume 281 of *Lecture Notes in Mathematics*, pages 66–105. Springer, 1972.

[Kel74] G.M. Kelly. On clubs and doctrines. volume 420 of *Lecture Notes in Mathematics*, pages 181–256. Springer, 1974.

[Kel92] G.M. Kelly. On clubs and data-type constructors. In M.P. Fourman, P.T. Johnstone, and A.M. Pitts, editors, *Applications of Categories in Computer Science*, volume 177 of *London Mathematical Society Lecture Notes*, pages 163–190. Cambridge University Press, 1992.

[KL80] G.M. Kelly and M.L. Laplaza. Coherence for compact closed categories. *Journal of Pure and Applied Algebra*, 19:193–213, 1980.

[KML71] G.M. Kelly and S. Mac Lane. Coherence in closed categories. *Journal of Pure and Applied Algebra*, 1:97–140, 1971.

[Kni96] E. Knill. Conventions for quantum pseudocode. Technical Report LAUR-96-2724, Los Alamos National Laboratory, 1996.

[Kos03] J. Koslowski. A monadic approach to polycategories. *Electronic Notes in Theoretical Computer Science*, 69:193–218, February 2003.

[KSW97] P. Katis, N. Sabadini, and R.F.C. Walters. Bicategories of processes. *Journal of Pure and Applied Algebra*, 115:141–178, 1997.

[Lam68] J. Lambek. Deductive systems and categories I. syntactic calculus and residuated categories. *Mathematical Systems Theory*, 2(4):287–318, 1968.

[Lam69] J. Lambek. Deductive systems and categories (ii). In R. J. Hilton, editor, *Category Theory, Homology Theory and their Applications I*, volume 87 of *Lecture Notes in Mathematics*, pages 76–122. Springer, 1969.

[Lam05] J. Lambek. Bicategories in algebra and linguistics. In P. Ruet, editor, *Linear Logic in Computer Science*. 2005.

[Loa94] R. Loader. *Models of Lambda Calculi and Linear Logic: Structural, Equational and Proof-theoretic Characterisations*. PhD thesis, St Hugh's College, Oxford, 1994.

[LS86] J. Lambek and P. J. Scott. *Introduction to higher order categorical logic*, volume 7 of *Cambridge studies in advanced mathematics*. Cambridge University Press, 1986.

[LTdF04] O. Laurent and L. Tortora de Falco. Slicing polarized additive normalisation. In T. Ehrhard, J.-Y. Girard, P. Ruet, and P. Scott, editors, *Linear Logic in Computer Science*, volume 316 of *London Mathematical Society Lecture Notes*, pages 247–282. Cambridge University Press, 2004.

[Mak05] M. Makkai. The word problem for computads. 2005.

[Man80] Y. Manin. Computable and uncomputable. *Sovetskoye Radio*, 1980. In Russian.

[Mét94] F. Métayer. Homology of proof-nets. *Arch. Math. Logic*, 33:169–188, 1994.

[Mét01] F. Métayer. Implicit exchange in multiplicative proofnets. *Mathematical Structures in Computer Science*, 11(2):261–272, 2001.

[Mil01] R. Milner. Bigraphical reactive systems. In *Proceedings of the 12th International Conference on Concurrency Theory*, volume 2154 of *Lecture Notes in Computer Science*, pages 16–35, 2001.

[MJ04] R. Milner and O. H. Jensen. Bigraphs and mobile processes. Technical Report UCAM-CL-TR-580, University of Cambridge Computer Laboratory, 2004.

[ML63] S. Mac Lane. Natural associativity and commutativity. *Rice Univ. Studies*, 49:28–46, 1963.

[ML97] S. Mac Lane. *Categories for the Working Mathematician (2nd Ed.)*. Springer-Verlag, 1997.

[MOTW95] J. Maraist, M. Odersky, D. Turner, and P. Wadler. Call-by-name, call-by-value, call-by-need, and the linear lambda calculus. In *11th International Conference on the Mathematical Foundations of Programming Semantics*, 1995.

[MT03] H. G. Mairson and K. Terui. On the computational complexity of cut-elimination in linear logic. In *Proceedings of ICTCS 2003*, number 2841 in Lecture Notes in Computer Science, pages 23–36. Springer-Verlag, October 2003.

[MTHM97] R Milner, M. Tofte, R Harper, and D. MacQueen. *Definition of Standard ML (Revised)*. MIT Press, 1997.

[NC00] M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 2000.

[O'H91] P. W. O'Hearn. Linear logic and interference control: Preliminary report. In S. Abramsky, P. L. Curien, A. M. Pitts, D. H. Pitt, A. Poigné, and D. E. Rydeheard, editors, *Proceedings of the Conference on Category Theory and Computer Science*, volume 530 of *Lecture Notes in Computer Science*, pages 74–93. Springer-Verlag, 1991.

[Öme03] B. Ömer. *Structured Quantum Programming*. PhD thesis, TU Vienna, May 2003.

[PB00] A.K. Pati and S. L. Braunstein. Impossibility of deleting an unknown quantum state. *Nature*, 404:164–165, 2000.

[Per93] A. Peres. *Quantum Theory: Concepts and Methods*. Kluwer Academic, 1993.

[Per05] S. Perdrix. Quantum patterns and types for entanglement and separability. In *Proceedings of the 3rd International Workshop on Quantum Programming Languages*, Electronic Notes in Computer Science, 2005.

[PJH99] S. L. Peyton Jones and J. Hughes. The Haskell 98 Report, 1999.

[Pra65] D. Prawitz. *Natural Deduction. A Proof-Theoretic Study*. Almqvist & Wiksell, 1965.

[RB01] R. Raussendorf and H. J. Briegel. A one-way quantum computer. *Phys. Rev. Lett.*, 86:5188–5191, 2001.

[RBB02] R. Raussendorf, D. E. Browne, and H. J. Briegel. The one-way quantum computer - a non-network model of quantum computation. *Journal of Modern Optics*, 49:1299, 2002, arXiv.org:quant-ph/0108118.

[RBB03] R. Raussendorf, D. E. Browne, and H. J. Briegel. Measurement-based quantum computation with cluster states. *Physical Review A*, 68(022312), 2003, arXiv.org:quant-ph/0301052.

[Sab03] A. Sabry. Modelling quantum computing in haskell. In *Proceedings of the ACM SIGPLAN Workshop on Haskell*. ACM, 2003.

[See89] R. A. G. Seely. Linear logic, *-autonomous categories and cofree algebras. In *Conference on Categories in Computer Science and Logic*, volume 92 of *AMS Contemporary Mathematics*, pages 371–382. June 1989.

[Sel04] P. Selinger. Towards a quantum programming language. *Mathematical Structures in Computer Science*, 14(4):527–586, 2004.

[Sel05] P. Selinger. Dagger compact closed categories and completely positive maps. In *Proceedings of the 3rd International Workshop on Quantum Programming Languages*, 2005.

[Sel06] P. Selinger. Idempotents in dagger categories. In P. Selinger, editor, *Proceedings of the 4th International Workshop on Quantum Programming Languages*, 2006.

[Shi96] M. Shirahata. A sequent calculus for compact closed categories, 1996.

[Sho97] P. W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J.Sci.Statist.Comput.*, 26(5), 1997.

[Sme01] S. Smets. *The Logic of Physical Properties in Static and Dynamic Perspective*. PhD thesis, Vrije Universiteit Brussel, May 2001.

[Spe] R. W. Spekkens. In defense of the epistemic view of quantum states: a toy model. *Phys. Rev. A*. (to appear).

[SS02] V. Sassone and P. Sobocinski. Deriving bisimulation congruences: a 2-categorical approach. *Electronic Notes in Theoretical Computer Science*, 68(2), 2002.

[SS04] V. Sassone and P. Sobocinski. Congruences for contextual graph-rewriting. Technical Report RS-04-11, BRICS, 2004.

[SU06] M. H. Sørensen and P. Urzyczyn. *Lectures on the Curry-Howard Isomorphism*, volume 149 of *Studies in Logic and the Foundations of Mathematics*. Elsevier, 2006.

[SZ00] J. Sanders and P. Zuliani. Quantum programming. In *Mathematics of Program Construction*, volume 1837 of *LNCS*, pages 80–99. Springer, 2000.

[Sza75] M.E Szabo. Polycategories. *Comm. Algebra*, 3:663–689, 1975.

[Tak87] G. Takeuti. *Proof Theory*. North-Holland Publishing, 1987.

[Tan97] A. M. Tan. *Full Completeness For Models Of Linear Logic*. PhD thesis, Cambridge University, 1997.

[Ter04] K. Terui. Proof nets and boolean circuits. In *Proceedings of the 19th Annual IEEE Symposium on Logic in Computer Science: LICS 2004*. IEEE Computer Society, 2004.

[Tro91] A. S. Troelstra. *Lectures on Linear Logic*, volume 29 of *CSLI Lecture Notes*. Centre for the Study of Languages and Information, Stanford, 1991.

[Vel88] Y. Velinov. An algebraic structure for derivations in rewriting systems. *Theoretical Computer Science*, 57:205–224, 1988.

[VP98] V. Vedral and M.B. Plenio. Entanglement measures and purification procedures. *Phys. Rev. A*, 57:1619–1633, 1998, arXiv:quant-ph/9707035.

[VPRK97] V. Vedral, M. B. Plenio, M. A. Rippin, and P. L. Knight. Quantifying entanglement. *Phys. Rev. Lett.*, 78(12):2275–2279, 1997.

[ZZHE93] M. Zukowski, A. Zeilinger, M. A. Horne, and A. K. Ekert. Event ready detectors bell experiment via entanglement swapping. *Phys. Rev. Lett.*, 71(26):4287, 4290 1993.

Index

$(-)^*$, 22
$(\cdot)^\dagger$, 27, 41
$(\cdot)^\perp$, 73
$(\mathfrak{P}, b)$, 140
$<_{\mathrm{in}(\cdot)}$, 98
$<_{\mathrm{out}(\cdot)}$, 99
$=_\beta$, 119
E_{ij}, *see* measurement calculus, entanglement operator
$F\mathcal{A}$, free compact closed category, 32
$G +_S H$, 98
R_β, 119
$[f]$, homotopy class of f, 115
$\Gamma - v$, 91
Γ/F, 110
$\Gamma_{\lhd e}$, 88
$\Gamma_{\rhd v}$, 90
$\mathfrak{R}$, 67
Ψ, canonical embedding, 105
Ψ_x, canonical embedding, 109
β-reduction
  for PN($\mathcal{A}$), 119
  for **mCQL** proof-nets, 56
β_i, *see* Bell basis
$\mathrm{cod}\,\Gamma$, 98
$[\![\cdot]\!]$, interpretation map
  for **mCQL** proof-nets, 57
  for **mCQL** proofs, 48
$\mathrm{dom}\,\Gamma$, 98
$\simeq$, 111
$\mathfrak{CU}$, 148
$\mathfrak{J}(\alpha)$, 143
$\ulcorner\cdot\urcorner$, $\llcorner\cdot\lrcorner$, 24
∂G, boundary of graph G, 90
$_i{}^k\circ_j$, 82
$\prec$, 87
$\sqsubseteq$, 87
$\mathrm{in}(\cdot)$, 87
$\mathrm{out}(\cdot)$, 87
f^τ_σ, *see* polyarrow, permutation
$s \bullet -$, scalar multiplication, 29
$\xrightarrow[\beta]{}$, 119

$\mathcal{A}$-labellable circuit, 105
$\mathcal{A}$-labelled circuit, 105
$\mathcal{A}$-labelling, 105
absorption lemma, 24
acyclic graph, 87
$\mathrm{Arr}_{\mathcal{A}}$, 104
associativity
  generalised, 83
  in a polycategory, 82
atomic link, 118
atomic proof-net, 118

Bell basis, 6
Bell state, 6, 39
boundary, 90
boundary node, 90
breaking a vertex, 91

canonical trace, 28
categorical quantum mechanics, 35
characterisation lemma, 62
Circ, 99, 100
Circ($\mathcal{A}$), 105
Circ($\mathcal{A}$)$^+$, 105
Circ$_{\mathrm{x}}(\mathcal{A})$, 109
Circ$^+_{\mathrm{x}}(\mathcal{A})$, 109
circuit, 98
Circ$^+$, 101
Circ$_{\mathrm{x}}(\mathcal{A})/\simeq$, 115
coequalisers in $\mathcal{G}_\partial$, 95
colimits in $\mathcal{G}_\partial$, 97
ComCl, 21, 41
commutativity conditions in polycategories, 83
compact closed category, 21
compact closed functor, 21
complete increasing chain, 93
composite system, 39

compositional cut, 26
compositionality lemma, 25
coname, 24
    partial, 26
confluence of β-reduction in PN($\mathcal{A}$), 119
connected graph, 87
contractible, 109
contraction, 10
contraction of circuits, 110
contractum along F, 110
coproducts in $\mathcal{G}_\partial$, 96
Curry-Howard isomorphism, 2
cut rule, 11
cut-elimination, 11
    for **mCQL**, 50
    for **mCQL** proof-nets, 57

dagger category, 41
Danos-Regnier criterion, 73
degree of a vertex, 87
$\dim_A$, 30
dimension, 30
directed graph, 87
    with circles, 88
directed path, 87
double gluing, 70, 71
dual, 19

elementary circuit, 99
EMC normal form, 139
endomorphisms, 31
entangled state, 67, 68, 70
    maximally, 70
entanglement, 67
entanglement swapping, 65
entangling sequence, 141
EPR state, 6
equivalence of categories, 18
even circuit, 99
evolution of quantum system, 6, 37, 39
extended $\mathcal{A}$-labelling, 109
extended labellings, 109

faithfulness
    of **mCQL** proof-net semantics, 64
FDHilb, 15, 37
formula
    of **mCQL**, 46
    of **MLL**, 72
free compact closed category
    generated by a category, 32
    generated by a polycategory, 104
free models of quantum mechanics, 40, 43
full completeness
    for **mCQL** proof-nets, 65
functor between polycategories, 85
fusing edges of a graph, 90

$\mathcal{G}$, 94
$\mathcal{G}_\partial$, 94
generalised proof-net, 117
graph, 87, 94
    with boundary, 90
    with circles, 94
graph with boundary, 94
Grph, 94

Hilbert-Schmitt norm, 31
homomorphism
    of graphs, 94
    of graphs with boundary, 94
    of graphs with circles, 94
homomorphism of circuits, 99
homotopy, 108
homotopy equivalence, 109
    of $\mathcal{A}$-labelled circuits, 111
    of graphs, 109

in-degree, 87
inference rules
    for $\mathbf{mCQL}_1$, 49
    for **mCQL**, 47
    for **MLL**, 73
Int construction, 28
interior, 90
interior node, 90
InvCat, 41
InvComClCat, 41
involution, 32
involutive categories, 41

Kelly-Laplaza theorem, 32

linear logic, 10
link, 55
    for PN($\mathcal{A}$) proof-nets, 118
loops, 31

maximal contraction, 110
maximally entangled state, 69, 70

mCQL$_1$
inference rules, 49
mCQL$_1$semantics, 49
mCQL$_1$, 49
mCQL$_2$, 49
mCQL
faithfulness, 64
formula, 46
inference rules, 47
semantics, 48
sequent calculus, 46
sequentialisation of proof-nets, 63
translation from sequents to proof-nets, 59
mCQL, 45
full completeness, 65
proof, 46
proof-net, 55
measurement, 37
non-degenerate, 37
measurement actions, 36
measurement calculus, 137
branch, 138, 140
corrections, 138
dependent corrections, 138
dependent measurements, 138
entanglement operator, 138
measurements, 138
pattern, 138
signals, 138
MLL
formula, 72
inference rules, 73
proof-net, 73
sequent calculus, 73
MLL, 10, 67, 72
proof-net, 73
Mon, 18
monoidal category, 16
symmetric, 16
traced, 28
monoidal functor, 17
from a polycategory, 86
strict, 18
strong, 18
symmetric, 18
monoidal natural transformation, 18
multi-cut, 82
multiplicative categorical quantum logic, 45
multiplicative linear logic, 10, 67, 72
multiplicative quantum mechanics, 36

name, 24
partial , 26
natural transformation between polycategory functors, 85
no-cloning theorem, 6, 135
no-deleting theorem, 7

observable, 7
one-way model of quantum computation, 137
out-degree, 87

pattern, *see* measurement calculus
peripheral node in a graph, 93
PN($\mathcal{A}$) proof-net, 117
PN($\mathcal{A}$), 117
polyarrow, 82
permutation, 83
polycategory, 82
compact symmetric, 82
with multicut, *see* polycategory, compact
positive circuit, 101
positive signed set, 101
prime circuit, 99
proof-net, 11
for **mCQL**, 55
for **MLL**, 73
for **mCQL**, 55
for **MLL**, 73
generalised, 117
proof-structure, 73

quantum bit, 6
quantum measurement, 37
quantum measurements, 7, 36
quantum mechanics
categorical, 35
categorical presentation, 39
Hilbert space, 5
multiplicative, 36
quantum state, 5, 37, 39
quantum teleportation, 8
qubit, 6

Rel, 15

scalars, 29

Schmidt coefficients, 68
Schmidt decomposition, 68
SComCl, 41
semantics
    for $\mathbf{mCQL}_1$ sequents, 49
    for **mCQL** sequents, 48
    of **mCQL** proof-nets, 57
separable state, 68, 70
separation lemma, 60
sequent calculus
    for **mCQL**, 46
    for **MLL**, 73
sequentialisation, 63
shuffle, 111
signals, *see* measurement calculus
signed set, 32, 100
simple circuit, 99
skeleton, 103
SMon, 18
soundness
    of β-reduction for **mCQL** proof-nets, 59
    of cut-elimination in **mCQL**, 55
splitting an edge in a graph, 88
standard order on $R + S$, 100
state space, 5, 36, 39
strict monoidal category, 16
strict monoidal functor, 18
strong monoidal functor, 18
strong normalisation of β-reduction in PN($\mathcal{A}$), 122
strongly compact closed category, 27
subject reduction for PN($\mathcal{A}$), 119
superposing axiom, 28
switching, 73
symmetric monoidal category, 16
symmetric monoidal functor, 18

teleportation, *see* quantum teleportation, 146
termination of β-reduction in PN($\mathcal{A}$), 119
topologically sorted, 141
trace, 28

unentangled state, 70
unitary evolution, 37, 39

vanishing axiom, 28

weakening, 10
yanking axiom, 28

zig-zag in a graph, 93