Computing Science Group

Geometry of abstraction in quantum computation

Dusko Pavlovic
Oxford University and Kestrel Institute

CS-RR-09-13

Oxford University Computing Laboratory
Wolfson Building, Parks Road, Oxford, OX1 3QD

Geometry of abstraction in quantum computation

Dusko Pavlovic
Oxford University and Kestrel Institute

Abstract

Quantum algorithms are sequences of abstract operations, performed on non-existent computers. They are in obvious need of categorical semantics. We present some steps in this direction, following earlier contributions of Abramsky, Coecke and Selinger. In particular, we analyze function abstraction in quantum computation, which turns out to characterize its classical interfaces.

Some quantum algorithms provide feasible solutions of important hard problems, such as factoring and discrete log (which are the building blocks of modern cryptography). It is of a great practical interest to precisely characterize the computational resources needed to execute such quantum algorithms. There are many ideas how to build a quantum computer. Can we prove some necessary conditions? Categorical semantics help with such questions. We show how to implement an important family of quantum algorithms using just abelian groups and relations.

1 Introduction

What do quantum programmers do? They do a variety of things, but there is a "design pattern" that they often follow, based on the *Hidden Subgroup Problem (HSP)* [26, 28, sec. 5.4]. Shor's factoring and discrete log algorithms [37] are examples of this pattern, as well as Hallgren's algorithm for the Pell equation [15]. They all provide an exponential speedup with respect to the best known classical algorithms. The simplest member of the family is Simon's algorithm for period finding [38], which we use as the running example. The other HSP algorithms only differ in "domain specific" details, but yield to the same semantics.

The input for Simon's algorithm is an arbitrary function $f : \mathbb{Z}_2^m \to \mathbb{Z}_2^n$, where $(\mathbb{Z}_2, \oplus, 0)$ is the group with two elements, and $\oplus$ is the "exclusive or" operation. The task is to find the period of f, if it exists, i.e. a bitstring $c \in \mathbb{Z}_2^m$ such that $f(x \oplus c) = f(x)$ for all $x \in \mathbb{Z}_2^m$. For simplicity, let us assume that there is exactly one such c, as the discussion of the other cases does not bring in anything essential.

Since f is arbitrary, one cannot ascertain that a bitstring c is a solution without computing the value of $f(x)$ for every $x \in \mathbb{Z}_2^m$. But a quantum computer can compute all such values at once! This is called *quantum parallelism*, and is one of the first things explained to quantum programmers' apprentices [28, sec. 1.4.2].

Mathematically speaking, the main capability of a quantum computer is that it can evaluate unitary operators. If the inputs of a function are represented as the basis vectors of a Hilbert space, and the function itself is captured as a unitary operator over it, then the quantum computer can compute all values of the function at once, by evaluating this unitary over a suitably generated combination of the basis vectors. Simon's algorithm shows how to extract the information about the period of the function from the projections of the resulting mixture.

But how do we represent a function $f : \mathbb{Z}_2^m \to \mathbb{Z}_2^n$ by a unitary operator? For an involutive function $g : B \to B$, the answer is easy: define $U_g : \mathbb{C}^B \longrightarrow \mathbb{C}^B$ by setting $U_g|b\rangle = |g(b)\rangle$, where $|b\rangle \in \mathbb{C}^B$ are the basis vectors indexed by $b \in B$. The fact

that U_g is unitary follows from $g \circ g = \mathrm{id}_B$. For a general $f : \mathbb{Z}_2^m \to \mathbb{Z}_2^n$, first define a corresponding involution f', and then extract the unitary U_f:

$$\frac{\frac{f : \mathbb{Z}_2^m \longrightarrow \mathbb{Z}_2^n : x \mapsto f(x)}{f' : \mathbb{Z}_2^{m+n} \longrightarrow \mathbb{Z}_2^{m+n} : x, y \mapsto x, y \oplus f(x)}}{U_f : \mathbb{C}^{\mathbb{Z}_2^{m+n}} \longrightarrow \mathbb{C}^{\mathbb{Z}_2^{m+n}} : |x, y\rangle \mapsto |x, y \oplus f(x)\rangle}$$

where the basis vectors $|x, y\rangle$ of $\mathbb{C}^{\mathbb{Z}_2^{m+n}}$ are indexed by the bitstrings x of length m concatenated with the bitstrings y of length n. The values of the function f are recovered from $U_f|x, 0\rangle = |x, f(x)\rangle$.

The other conceptual component of Simon's algorithm, and of all HSP-algorithms, is a standard application of transform theory [39]: transform the inputs into another domain, where the computation is easier, compute the outputs there, and then transform them back[1]. In our special case, U_f is thus precomposed and postcomposed with a suitable version of the Fourrier transform, which for $\mathbb{Z}_2$ boils down to the Hadamard-Walsh tranform $H^{\otimes m}|z\rangle = \sum_{x \in \mathbb{Z}_2^m} (-1)^{x \cdot z}|x\rangle$. Here $x \cdot z$ denotes the inner product in $\mathbb{Z}_2^m$, and we ignore the renormalizing factor $2^{-\frac{m}{2}}$. This transform is applied to the first m arguments of U_f, to generate the desired superposition of all inputs of f. The quantum computer thus computes the followsing vector[2]:

$$\begin{aligned} Simon &= (H^{\otimes m} \otimes \mathrm{id}) U_f (H^{\otimes m} \otimes \mathrm{id})\, |0, 0\rangle \\ &= \sum_{z, x \in \mathbb{Z}_2^m} (-1)^{x \cdot z} |z, f(x)\rangle \end{aligned}$$

When we measure the first component of this vector, it collapses to a single $|z\rangle$, i.e. we get $\gamma_z = |z\rangle \otimes \sum_{x \in \mathbb{Z}_2^m} (-1)^{x \cdot z}\, |f(x)\rangle$. By assumption, there is exactly one $c \in \mathbb{Z}_2^m$ such that $f(x \oplus c) = f(x)$ holds for all x. The coefficient of each of the basis vectors $|z, f(x)\rangle = |z, f(x \oplus c\rangle$ is thus $\gamma_z^x = (-1)^{x \cdot z} + (-1)^{(x \oplus c) \cdot z} = (-1)^{x \cdot z}\,(1 + (-1)^{c \cdot z})$. It follows that $(\forall x \in \mathbb{Z}_2^m .\, \gamma_z^x \neq 0) \iff c \cdot z = 0$. Each time that we run the algorithm, we can thus extract a linear equation in c. After m runs, we can thus compute c. (The probability that at some step $k \leq m$ we may get an equation dependent on the previous ones is 0, because z are chosen randomly, and the measure of every proper linear subspace of $\mathbb{Z}_2^m$ is 0.) On the other hand, in order to convince ourselves classically that c is the period of f, we should to compute all values f, which requires 2^m steps, since f is an arbitrary function.

The core of Shor's factoring algorithm follows the same pattern, adapted for $f : \mathbb{Z}_k \to \mathbb{Z}_k$, where $f(x) = a^x \mod k$. The factored integer is k, and a is randomly selected to be tested for common factors with it, which can be derived by finding a period of f.

Summary of the paper. A program generally describes a family of computations over a family of input data. The various input data to be computed with are denoted by variables. E.g., the polynomial $x^2 + x + 2$ can be construed as a program, describing the family of computations that can be performed for the various values of x. It is tacitly assumed that the possible values of x can be copied, so that one copy can be substituted for each occurrence of x in the polynomial $x^2 + x$; and that these data can also be deleted, if the polynomial is just 2, and x does not occur in it.

The first problem with quantum programming is that quantum data cannot be manipulated in this way: it is a fundamental property of quantum states that they generally cannot be copied [41, 11], or even deleted [29, 2]. So how do we write quantum programs? In particular, given a program $f(x)$ for a function f, what kind of a program transformation leads to the quantum program $U_f|x, y\rangle$, that we used to specify the unitary U_f above? This question is analyzed and answered in sections 3 and 4. It turns out that the needed copying and deleting operations are closely related with the abstraction.

On the other hand, copying, deleting and abstraction capabilities can be viewed as the characteristics of classical computation. In a quantum computer, a structure that supports copying, deleting and abstraction can be construed as its classical interface. This is what we call a *classical structure*. An early analysis of this structure was in [7]. In the meantime, there are several versions, and many applications [8, 31, 12]. In recent work, Coecke [9] uses the term *basis structures* for the same concept, because a classical structure over a finitely dimensional Hilbert space precisely correspond to a choice of a basis [10], and can be viewed as a purely categorical, element-free version of this notion. While the simple basis intuitions are attractive, I stick

[1] E.g., Laplace's transform maps a differential equation into a polynomial equation over the field, generated by the convolution ring in which the original equation was stated [32]. The solutions of the polynomial equation are then mapped back by the inverse Laplace transform.

[2] We ignore the renormalizing factors throughout.

here with the original terminology. One reason is that the correspondence of classical structures and the induced bases is not always as simple as it is in the category of finitely dimensional Hilbert spaces [31], and it is useful to keep the distinction. A more important reason is that classical structures express the fact that *classicality is relative* as an algebraic structure. The fact that classical data with respect to one classical structure may be entangled with respect to another one is the fundamental feature of quantum computation. This is usually captured through change of basis. Classical structures provide an algebraic framework for such transforms. This is summarized in section 5.

The final step of the described algorithm pattern, measurement, is modeled in section 6. The resulting categorical semantics is supported not only by the standard Hilbert space model, but also by non-standard models. We spell out a relational interpretation, based on [9, 31]. In particular, Simon's algorithm turns out to have an effective relational implementation, using an abelian group as the computational resource supplying the power of a quantum computer.

Section 2 provides a brief summary of the basic semantical prerequisites, notations and terminology.

2 Preliminaries

2.1 Monoidal categories

We assume that the reader has some understanding of the basic categorical concepts and terminology [27], and work with symmetric monoidal categories $(\mathcal{C}, \otimes, I)$ [17, 16].

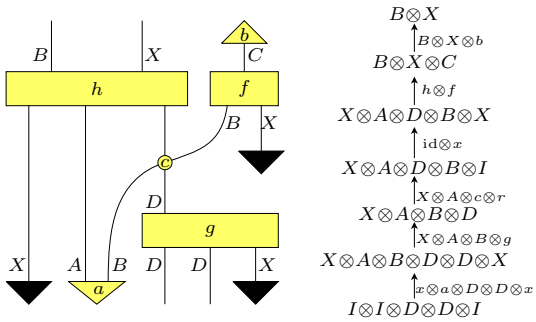
Strictness. For simplicity, and without loss of generality, we tacitly assume that each of our monoidal categories is *strictly associative and unitary*, i.e. that the objects form a monoid in the usual sense. This causes no loss of generality because every monoidal category is equivalent to a strictly associative and unitary one, along a monoidal equivalence. But note that the tensor symmetry cannot be "strictified" without essentially changing the category; the canonical isomorphisms $A \otimes B \xrightarrow{c} B \otimes A$ are thus generally *not* identities.

On the other hand, just like the tensors, we strictify functors: a *monoidal* functor F is always assumed to be strict, i.e. it preserves the monoidal structure on the nose: $F(A \otimes B) = FA \otimes FB$ and $FI = I$.

The arrows from I are sometimes called *vectors*, or *elements*. The abstract "vector spaces" are thus written $\mathcal{C}(X) = \mathcal{C}(I, X)$. When confusion is unlikely, we elide the tensor symbol and write XAf instead of $X \otimes A \otimes f$.

2.1.1 String diagrams

Calculations in monoidal categories are supported by a simple and intuitive graphical language: the string diagrams. This language has its roots in Penrose's diagrammatic notation [33], and it has been formally developed in categorical *coherence theory*, and in particular in Joyal and Street's *geometry of tensor calculus* [16]. The objects are drawn as strings, and the morphisms as boxes attached on these strings. One can think that the information flows through the strings, and is processed in the boxes. A direction of this flow is chosen by convenience. We shall assume that the information flows up, so that the strings at the bottom of a box denote the domain of the corresponding morphism; the threads at the top the codomain. Drawing the strings A and B next to each other represents $A \otimes B$; similarly with the boxes. Drawing a thread from one box to another is denotes the composition of the corresponding morphisms.



One of the salient features if this notation is that the associativity is implicit, and automatic, both of the tensor and of the composition. The tensor symmetry $c : B \otimes D \longrightarrow D \otimes B$ is denoted above by a circle. The circle is usually omitted, so that symmetry boils down to crossing the strings. The identity morphisms are the "invisible boxes", that can be placed on any thread. The tensor unit I is the "invisible thread", that can be added to any diagram. This means that a box representing a vector $a \in \mathcal{C}(I, AB)$ does not have any visible threads coming in from below. This is often emphasized by reducing the bottom of such a box to a point: e.g., the vector $I \xrightarrow{a} AB$ is denoted by a triangle. The box representing a covector $b \in \mathcal{C}(C, I)$ does not have any visible threads coming out, and boils down to a triangle pointing up. The black triangles denote the vector indeterminates $I \xrightarrow{x} X$, freely adjointed to monoidal categories to form polynomials. Such polynomial constructions will be discussed in Sec. 3.

2.1.2 Monoids and comonoids

A monoid in a monoidal category is a pair of arrows $X \otimes X \xrightarrow{\nabla} X \xleftarrow{\perp} I$ such that

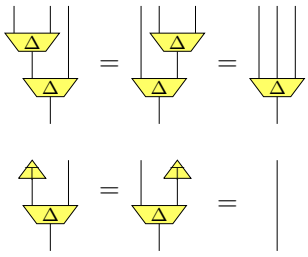
$$\nabla \circ (\nabla \otimes X) = \nabla(X \otimes \nabla)$$
$$\nabla \circ (\perp \otimes X) = \nabla \circ (X \otimes \perp) = \mathrm{id}_X$$

When the tensor is the cartesian product, this captures the usual notion of monoid.

A comonoid in a monoidal category is dual to a monoid: it is a pair of arrows $X \otimes X \xleftarrow{\Delta} X \xrightarrow{\top} I$ such that

$$(\Delta \otimes X) \circ \Delta = (X \otimes \Delta) \circ \Delta$$
$$(\top \otimes X) \circ \Delta = (X \otimes \top) \circ \Delta = \mathrm{id}_X$$

In **string diagrams**, we draw the monoid evaluations as trapezoids pointing up, whereas their units are little triangles pointing down. The comonoids are represented by the trapezoids and the little triangles in the opposite directions. E.g., the comonoid laws correspond to the following graph transformations



A monoid is *commutative* if $\nabla \circ c_{XX} = \nabla$. A comonoid is commutative if $c_{XX} \circ \Delta = \Delta$. In string diagrams, this means that the value of the output of ∇ does not change if the strings that come into it cross; and that the output of Δ does not change if the strings coming out of it cross.

2.1.3 Cartesian categories

A monoidal category $(\mathcal{C}, \otimes, I)$ is *cartesian* when it comes with natural transformations

$$X \otimes X \xleftarrow{\delta_X} X \xrightarrow{!_X} I$$

which make every object X into a comonoid. The naturality of this structure means that every morphism $X \xrightarrow{f} Y$ in $\mathcal{C}$ is a comonoid homomorphism. It is easy to see that this makes the tensor $X \otimes Y$ into a product $X \otimes Y$, such that any pair of arrows $A \xrightarrow{g} X$ and $A \xrightarrow{h} Y$ corresponds to a unique arrow $A \xrightarrow{\langle g,h \rangle} A \times B$, and the tensor unit I into the final object 1, with a unique arrow from each object. Cartesian structure is thus written in the form $(\mathcal{C}, \times, 1)$.

2.1.4 Monads and comonads

A *monad* on a category $\mathcal{C}$ can be defined as a functor $T : \mathcal{C} \longrightarrow \mathcal{C}$ together with a monoid structure $TT \xrightarrow{m} T \xleftarrow{h} \mathrm{Id}$ in the category of endofunctors on $\mathcal{C}$. With the corresponding monoid homomorphisms, monads form a category on their own [3]. Dually, *comonads* on $\mathcal{C}$ can be defined as comonoids in the category of endofunctors over $\mathcal{C}$, and accomodate similar developments.

The categories of algebras for a monad and coalgebras for a comonad, and in particular the Kleisli and the Eilenberg-Moore constructions that will be used below, are presented in detail in [27, 3], and in many other books.

The following observation is the starting point for most of the constructions in this paper. The proof is left as an easy exercise.

Proposition 2.1 *Every (co)monoid X in a monoidal category $\mathcal{C}$ induces a (co)monad $X \otimes (-) : \mathcal{C} \longrightarrow \mathcal{C}$. The corresponding Kleisli category $\mathcal{C}_{[X]}$ is monoidal if and only if the (co)monoid X is commutative.*

More precisely, the category of monoids in a monoidal category $\mathcal{C}$ is equivalent with the category of monads T on $\mathcal{C}$ such that $T(A \otimes B) = T(A) \otimes B$ and moreover $h_B = h_I \otimes B$ and $m_B = m_I \otimes B$ hold for all $A, B \in \mathcal{C}$. The dual statement holds for comonoids and comonads.

2.1.5 Convolution and representation

Any monoid $(X, \triangledown, \perp)$ in a monoidal category $(\mathcal{C}, \otimes, I)$ induces the ordinary monoid $(\mathcal{C}(X), \bullet, \perp)$, whose operation

$$a \bullet b \quad = \quad \triangledown \circ (a \otimes b) \tag{1}$$

is often called *convolution*. A Cayley representation (or Yoneda embedding) of the monoid $(X, \triangledown, \perp)$ is a map

$$\begin{array}{rcl} \widehat{(-)} : \mathcal{C}(X) & \longrightarrow & \mathcal{C}(X, X) \\ \left(I \xrightarrow{a} X\right) & \longmapsto & \left(X \xrightarrow{a \otimes X} X \otimes X \xrightarrow{\triangledown} X\right) \end{array} \tag{2}$$

furthermore represents the vectors $a \in \mathcal{C}(X)$ as endomorphisms $\widehat{a} \in \mathcal{C}(X, X)$.

Lemma 2.2 *(Cayley, Yoneda) The Cayley representation is a monoid isomorphism between the convolution monoid $(\mathcal{C}(X), \bullet, \perp)$ and the monoid $(\mathrm{Nat}(X, X), \circ, \mathrm{id}_X)$ of* natural *endomorphisms*

$$\mathrm{Nat}(X, X) \quad = \quad \{f \in \mathcal{C}(X, X) \mid \forall ab \in \mathcal{C}(X).\ f \circ (a \bullet b) = (f \circ a) \bullet b\}$$

A comonoid structure on X induces a convolution monoid on $\mathcal{C}(X, I)$, with $c \bullet d = (c \otimes d) \circ \triangle$, and with a similar Cayley representation. In general, a convolution monoid can be defined over any hom-set $\mathcal{C}(X, Y)$, where X is a comonoid and Y a monoid, by setting $f \bullet g = \triangledown_Y \circ (f \otimes g) \circ \triangle_X$.

Scalars. The canonical isomorphism $I \otimes I \cong I$ makes the tensor unit I of $\mathcal{C}$ into a commutative monoid and comonoid; the tensor associativity is the associativity law of this (co)monoid; the tensor commutativity makes the (co)monoid commutative; the coherence conditions tell that this is the only (co)monoid structure on I. The convolution monoid $(\mathcal{C}(I,I), \bullet, \mathrm{id}_I)$ is the abstract *scalar algebra* of the monoidal category $\mathcal{C}$. The coherence conditions imply that there is only one monoid structure on I, hence $s \bullet t = s \circ t = s \otimes t$ holds for all scalars $s, t \in \mathcal{C}(I,I)$.

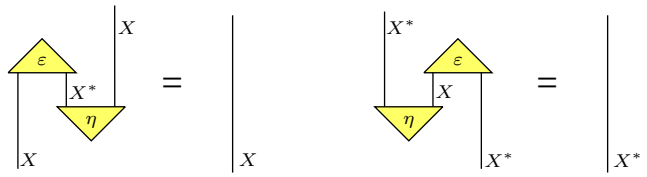
Abusing notation, the scalar action $\bullet : \mathcal{C}(I,I) \times \mathcal{C}(A,B) \longrightarrow \mathcal{C}(A,B)$ is defined by $s \bullet f = s \otimes f$. If the tensor unit I is not strict, then $s \otimes f$ needs to be precomposed by $A \cong I \otimes A$ and postcomposed by $I \otimes B \cong B$.

2.2 Duals with daggers

2.2.1 Dualities

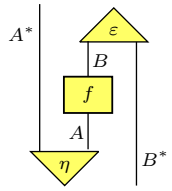
A *duality* structure in a monoidal category $\mathcal{C}$ consists of two objects X and X^* and two arrows, the pairing $X \otimes X^* \xrightarrow{\varepsilon} I$ and the copairing $I \xrightarrow{\eta} X^* \otimes X$, such that

$$(\varepsilon \otimes X)(X \otimes \eta) = X \qquad\qquad (X^* \otimes \varepsilon)(\eta \otimes X^*) = X^*$$



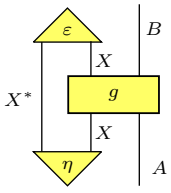
A duality structure is written $(\eta, \varepsilon) : X \dashv X^*$. Note that $X^{**} = X$, because $(c\eta, \varepsilon c) : X^* \dashv X$ is also a duality structure. If every object $X \in \mathcal{C}$ has a chosen duality structure, then such choices induce a *duality functor* $* : \mathcal{C}^{op} \longrightarrow \mathcal{C}$, which maps $A \xrightarrow{f} B$ to

$$f^* : B^* \xrightarrow{\eta B^*} A^* A B^* \xrightarrow{A f B^*} A^* B B^* \xrightarrow{A^* \varepsilon} A^*$$



Using a duality $(\eta, \varepsilon) : X \dashv X^*$, the abstract trace operators $\mathrm{Tr}_X^{AB} : \mathcal{C}(XA, XB) \longrightarrow \mathcal{C}(A,B)$ can be defined as follows:

$$\mathrm{Tr}_X^{AB} g : A \xrightarrow{\eta_X A} X^* X A \xrightarrow{X^* g} X^* X B \xrightarrow{\varepsilon_{X^*} B} B$$



2.2.2 Dagger-monoidal categories

A *dagger* over a category $\mathcal{C}$ is an involutive ioof $\ddagger : \mathcal{C}^{op} \longrightarrow \mathcal{C}$. In other words, it satisfies $A^\ddagger = A$ on the objects and $f^{\ddagger\ddagger} = f$ on the arrows. This very basic structure turns out to suffice for some crucial concepts.

Definition 2.3 *A morphism* $u \in \mathcal{C}(A,B)$ unitary *if* $u^\dagger \circ u = \mathrm{id}_A$ *and* $u \circ u^\dagger = \mathrm{id}_B$. *An endomorphism* $p \in \mathcal{C}(A,A)$ *is a* projector *if* $p = p^\dagger = p \circ p$. *A projector is* pure *if moreover* $\mathrm{Tr}_A^{II}(p) = \mathrm{id}_I$.

Remarks. Note that the abstract trace operators, given above, require a monoidal structure in $\mathcal{C}$. The interactions between the dagger with the monoidal structure, and in particular with the duals, has been recognized and analyzed in [1, 35, 36]. A *dagger-monoidal* category $(\mathcal{C}, \otimes, I, \dagger)$ is a dagger-category with a monoidal structure where all canonical isomorphisms, that form the monoidal structure, are unitary. When the monoidal structure is strict, this boils down to the requirement that the symmetry $c : A \otimes B \to B \otimes A$ is unitary.

In the **string diagrams**, the morphism $f^\dagger$ is represented by flipping the box f around its horizontal axis. The morphism boxes thus need to be made asymmetric to record this flipping: in [35], a corner of the box is filled; in [7], a corner is cut off.

2.2.3 Abstract conjugates and reals

Since the dagger and the duality functors $(-)^\dagger, (-)^* : \mathcal{C}^{op} \longrightarrow \mathcal{C}$ commute, their composite defines the *conjugation* ioof $(-)_* : \mathcal{C}^{op} \longrightarrow \mathcal{C}$, which maps f to $f_* = f^{*\dagger} = f^{\dagger *}$. In the category of complex Hilbert spaces, the conjugation ioof corresponds is induced by the conjugation of the complex numbers. In the category of real Hilbert spaces, it degenerates into the identity functor.

Definition 2.4 *A morphism* f *is said to be* real *if* $f = f_*$ *(or equivalently* $f^\dagger = f^*$*).*

Remarks. Pursuing the Hilbert space intuitions, the arrows f and $f^\dagger$ are sometimes thought of as each other's adjoints. On the other hand, in a completely different sense, the dual objects A and A^* are each other's adjoints, if the monoidal category is vewed as a bicategory with one object.

2.2.4 Inner products and entanglement

The dagger-monoidal structure has been proposed as a framework for categorical semantics of quantum computation [1, 35]. It turns out that this modes structure suffices for deriving many important notions:

- *inner product*

$$\begin{array}{rcl} \langle -|- \rangle_A \ : \ \mathcal{C}(A) \times \mathcal{C}(A) & \longrightarrow & \mathcal{C}(I) \\ (I \xrightarrow{a,b} A) & \longmapsto & \left(I \xrightarrow{a} A \xrightarrow{b^\dagger} I \right) \end{array} \tag{3}$$

- *partial inner product*

$$\begin{array}{rcl} \langle -|- \rangle_A^B \ : \ \mathcal{C}(A) \times \mathcal{C}(AB) & \longrightarrow & \mathcal{C}(B) \\ \left(I \xrightarrow{a} A, I \xrightarrow{b} AB \right) & \longmapsto & \left(I \xrightarrow{a} AB \xrightarrow{b^\dagger \otimes B} B \right) \end{array} \tag{4}$$

- *weakly entangled vectors* $\eta \in \mathcal{C}(A \otimes A)$, such that for all $a \in \mathcal{C}(A)$ holds

$$\langle a_* \mid \eta \rangle_A^A \quad = \quad a \tag{5}$$

Furthermore, an abstract version of strong entanglement can be defined as self-duality.

Definition 2.5 *A vector* $\eta \in \mathcal{C}(X \otimes X)$ *is said to be (strongly) entangled if* $(\eta, \eta^\dagger) : X \dashv X$ *is a duality, i.e. satisfies* $(\eta^\dagger \otimes X)(X \otimes \eta) = X = (X \otimes \eta^\dagger)(\eta \otimes X)$*, and thus* $X^* = X$.

Proposition 2.6 *For every object* X *in a dagger-monoidal category* $\mathcal{C}$ *holds (a)* $\iff$ *(b)* $\Longleftarrow$ *(c), where*

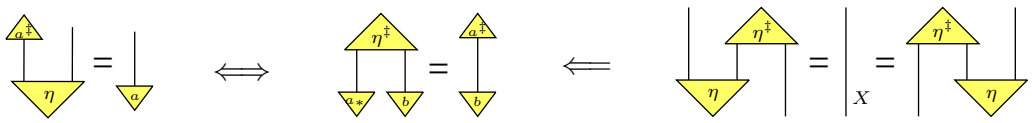
(a) $\eta \in \mathcal{C}(X \otimes X)$ *is weakly entangled*

(b) $\eta^\ddagger \in \mathcal{C}(X \otimes X, I)$ *internalizes the inner product, as* $\langle a|b\rangle = \eta^\ddagger \circ (a_* \otimes b)$

(c) $\eta \in \mathcal{C}(X \otimes X)$ *is strongly entangled.*

The three conditions are equivalent if I *generates* $\mathcal{C}$*, in the sense that whenever* $fa = ga$ *for all* $a \in \mathcal{C}(X)$*, then* $f = g$*.*

A **proof** can be conveniently built from transformations among the string diagrams of the conditions:



2.3 Notation and terminology

To describe relations on finite sets, we often find it convenient to use von Neumann's representation of ordinals, where $0 = \emptyset$ is the empty set, and $n = \{0, 1, \ldots, n-1\}$. Moreover, the pairs $\langle i, j\rangle \in n \times n$ are often abbreviated to $ij \in n \times n$.

When space is constrained and confusion unlikely, we often elide the tensors and write $AfXX$ instead of $A \otimes f \otimes X \otimes X$.

Ioofs and embeddings. Many categorical constructions lead to functors where the object part is the identity. They are often called Identity-Onthe-Objects-Functors. I call them *ioofs*. If the reader finds this abbreviation objectionable, she is welcome to unfold each of its occurrences, and read out the full phrase .

In a similar development, the functors that are full and faithful are often called Full-and-Faithful-Functors. I call them *embeddings*. The reader may notice that every functor can be factored into an ioof followed by an embedding.

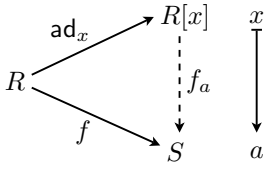
3 Polynomials and abstraction

In this section we formalize the program transformations needed to implement a classical function in a quantum computer. If a program is an arrow in a category, a program transformation is simply a functor out of it. But the problem with transforming a classical program into a quantum program is that classical data can be copied and deleted, whereas quantum data cannot. So the program transformation must map classical data to classical data, distinguished within a quantum universe. What does this mean? When the classical program $f'(x, y)$ was transformed into the corresponding quantum program $U_f|x, y\rangle$ in the Introduction, the classical inputs were denoted by the variables x, y, and mapped to the basis vector variables $|x, y\rangle$. The fact that the classical inputs can be copied and deleted was captured as a syntactical property of the variables.

If the data over which a program will compute are denoted by variables, then the program itself is a polynomial in some suitable algebraic theory. More precisely, a program is an *abstraction* over the as-yet-undetermined input data, and a computation is an *application* of the program. More generally, a program transformation can be viewed as a *substitution* into a polynomial. So we need functorial semantics of polynomial constructions, and of the abstraction and substitution operations. In the framework of cartesian (closed) categories, such a treatment goes back to Lambek and Scott's seminal work [22, 23]. It was extended to monoidal categories in [30]. Here we extend it to dagger-monoidal categories.

3.1 Polynomial constructions

Adjoining an indeterminate x to a ring R leads to the ring of polynomials $R[x]$. Its universal property is that every ring homomorphism $f : R \longrightarrow S$ extends to a unique ring homomorphism $f_a : R[x] \longrightarrow S$ for each choice of $a \in S$ to which x is mapped.



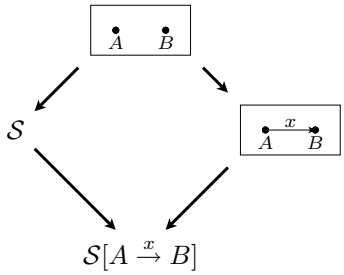
The same construction applies to other algebraic theories: e.g., one could form polynomial groups, or polynomial lattices. Categorically, for an arbitrary algebraic theory T, a polynomial T-algebra $A[x]$ can be viewed as the coproduct in the category of T-algebras of the T-algebra A and the free T-algebra over one generator, denoted x.

The polynomial construction also applies to algebraic structures over categories, such as cartesian, monoidal, or $*$-autonomous; polynomial categories can be built for any algebraic theory T over the category of categories. The polynomial category $\mathcal{S}[x]$ is then the free T-category obtained by freely adjoining a single generator x to the T-category $\mathcal{S}$; i.e. as the coproduct of $\mathcal{S}$ and the free T-category generated by x. However, categories are generated over graphs, rather than sets, so the question is what kind of a graph should x be. There seem to be two minimal choices:

(a) x is an object: a graph with one node and no edges; or

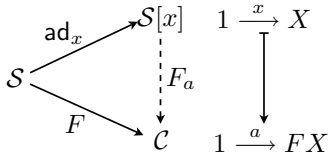
(b) x is an arrow: a graph with two nodes and an edge between them.

While case (a) leads to the constructions which do not involve the arrows, and thus largely boil down to the polynomial constructions of universal algebra, case (b) involves genuinely categorical aspects. These new aspects are isolated by assuming that *only* new arrows are adjoined to $\mathcal{S}$, and *no* new objects. More precisely, an indeterminate arrow $A \xrightarrow{x} B$ is freely adjoined between the extant objects A, B of $\mathcal{S}$. In other words, $\mathcal{S}[A \xrightarrow{x} B]$ can be viewed as the following pushout



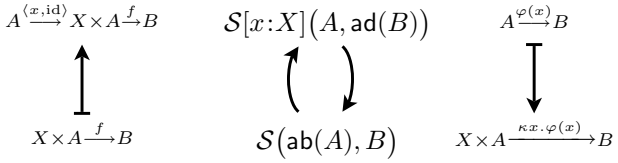
in the category of T-categories.

Lambek was the first to use polynomial categories in his interpretation of typed λ-calculus in cartesian closed categories [22]. The approach was elaborated in the book [23], from which categorical semantics branched in many directions. The terms containing a variable x of type X were represented as the arrows of the polynomial category $\mathcal{S}[x{:}X]$, built by adjoining to a cartesian closed category $\mathcal{S}$ an indeterminate arrow $1 \xrightarrow{x} X$, where X is an object of $\mathcal{S}$. The universal property of $\mathcal{S}[x:X]$ is the same as before: every structure preserving functor $F : \mathcal{S} \longrightarrow \mathcal{L}$ extends to a unique structure-preserving functor $F_a : \mathcal{S}[x] \longrightarrow \mathcal{L}$ by mapping $1 \xrightarrow{x} X$ to $1 \xrightarrow{a} FX$ in $\mathcal{L}$.



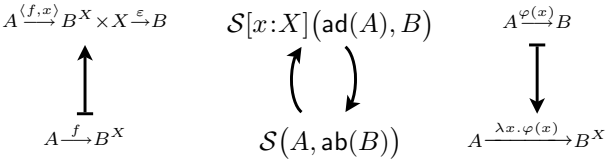
Just like a polynomial ring, the category $\mathcal{S}[x:X]$ can be constructed syntactically. However, the cartesian closed structure allows a more effective and more familiar presentation of $\mathcal{S}[x{:}X]$.

Theorem 3.1 [22, 23] *Let $\mathcal{S}$ be a cartesian category, $X \in \mathcal{S}$ an object and $\mathcal{S}[x{:}X]$ the free cartesian category generated by $\mathcal{S}$ and $1 \xrightarrow{x} X$. Then the inclusion functor* $\mathsf{ad} : \mathcal{S} \longrightarrow \mathcal{S}[x:X]$ *has a left adjoint, the* abstraction *functor* $\mathsf{ab} : \mathcal{S}[x:X] \longrightarrow \mathcal{S} \ : \ A \mapsto X \times A$



and $\mathcal{S}[x{:}X]$ is equivalent with the Kleisli category for the comonad $X \times (-) : \mathcal{S} \longrightarrow \mathcal{S}$.

When $\mathcal{S}$ is cartesian closed, then $\mathcal{S}[x{:}X]$ is cartesian closed too. The Kleisli category for the comonad $X \times (-) : \mathcal{S} \longrightarrow \mathcal{S}$ is isomorphic with the Kleisli category for the monad $(-)^X : \mathcal{S} \longrightarrow \mathcal{S}$. The abstraction functor can now be viewed as a right *adjoint of the inclusion* $\mathsf{ad} : \mathcal{S} \longrightarrow \mathcal{S}[x{:}X]$



This latter adjunction provides a categorical model of simply typed lambda-calculus.

Notion of abstraction. Function abstraction is what makes programming possible. The first example of program abstraction were probably Gödel's numberings of primitive recursive functions [14]. Gödel's construction demonstrated that recursive programs, specifying entire families of computations (of the values of a function for all its inputs), can be stored as data. Von Neumann later explicated this as the fundamental principle of computer architecture. Kleene, on the other side, refined the idea of program abstraction into the fundamental lemma of recursion theory: the s-m-n theorem [19]. Church, finally[3] proposed the formal operations of function abstraction and data application as the driving force of all computation [6]. This proposal became the foundation of functional programming. Lawvere's observation that Church's λ-abstraction could be interpreted as an adjunction transposition [24] was a critical step towards categorical semantics of computation. Theorem 3.1 spells out this observation in terms of polynomial categories. Besides the familiar λ-abstraction, which uses the right adjoint of the inclusion $\mathsf{ad} : \mathcal{S} \longrightarrow \mathcal{S}[x{:}X]$ to transpose a polynomial into a function which outputs functions

$$\frac{\varphi(x) : A \to B}{\lambda x.\varphi(x) : A \to B^X}$$

the theorem points to an analogous abstraction operation which uses the *left* adjoint to the inclusion $\mathsf{ad} : \mathcal{S} \longrightarrow \mathcal{S}[x{:}X]$, and transposes polynomials into *indexed* families of functions

$$\frac{\varphi(x) : A \to B}{\kappa x.\varphi(x) : X \times A \to B}$$

This form of abstraction does not require higher-order types, and lifts from cartesian to monoidal categories [30]. In the present paper, we extend such abstraction operations to monoidal categories with enough structure to support the basic forms of quantum programming. — In this way, the usual quantum programming constructions can be viewed as a form of functional programming in Hilbert spaces.

But what kind of functional programming is it?

The fundamental assumption of functional programming is that all data can be copied and deleted. Theorem 3.1 shows that this implies a canonical abstraction operation.

The fundamental assumption of quantum programming is that some data —the quantum data— cannot be copied or deleted; but they can be entangled. Entanglement is then developed into a powerful computational resource. In-between the data that can be copied and deleted, and the data that can be entangled, there is a rich structure of diverse abstraction operations, that we shall now explore. The idea is that quantum programming can be "semantically reconstructed" a set of techniques for combining and interfacing quantum entanglement and classical abstractions.

[3] Although Church's paper appeared three years earlier than Kleene's, Church's proposal is the final step in the conceptual development of function abstraction as the foundation of computation.

3.2 Abstraction in monoidal categories

Given a monoidal category $\mathcal{C}$ and a chosen object A in it, we want to freely adjoin a variable arrow $I \xrightarrow{x} A$ and build the polynomial monoidal category $\mathcal{C}[x:X]$. Like before, $\mathcal{C}[x:X]$ can be built syntactically, as the free symmetric monoidal category over the graph spanned by $\mathcal{C}$ and $I \xrightarrow{x} A$, factored by the equations between the arrows of $\mathcal{C}$. Although this is not a very effective description, it does show that the polynomial category $\mathcal{C}[x:X]$ can in this case be quite complicated[4]. Moreover, in contrast with the cartesian (closed) case, the inclusion $\mathsf{ad} : \mathcal{C} \longrightarrow \mathcal{C}[x:X]$ does not have an adjoint in general, and thus does not support abstraction. The task is now to extend the polynomial construction to support abstraction. We follow, refine and strengthen the results from [30].

Definition 3.2 *Let $\mathcal{C}$ be a monoidal category, and E a set of well typed equations between some polynomial arrows in $\mathcal{C}[x:X]$. A* monoidal extension *is the monoidal category $\mathcal{C}[x:X;E] = \mathcal{C}[x:X]/E$ obtained by imposing the equations E on $\mathcal{C}[x:X]$, together with all equations that make it into a monoidal category. Every monoidal extension comes with the obvious ioof $\mathsf{ad} : \mathcal{C} \longrightarrow \mathcal{C}[x:X;E]$.*

A substitution functor *between monoidal extensions is a (strict) monoidal ioof $F : \mathcal{C}[x:X;E] \longrightarrow \mathcal{C}[y:Y;D]$.*

We denote by $\mathsf{Ext}_{\mathcal{C}}$ the category of monoidal extensions of $\mathcal{C}$, with the substitution functors between them.

Definition 3.3 *A* (monoidal) abstraction *over a monoidal extension $\mathsf{ad} : \mathcal{C} \longrightarrow \mathcal{C}[x:X;E]$ is the adjunction $\mathsf{ab} \dashv \mathsf{ad}$ such that $\mathsf{ab}(A \otimes B) = \mathsf{ab}(A) \otimes B$, and the unit of the adjunction $h : \mathrm{Id} \longrightarrow \mathsf{ad} \circ \mathsf{ab}$ satisfies $h_A = x \otimes A$. We denote by $\mathsf{Abs}_{\mathcal{C}}$ the subcategory of $\mathsf{Ext}_{\mathcal{C}}$ spanned by the monoidal extensions that support abstraction.*

Notation and terminology. Since the abstraction notation $\mathsf{ab} \dashv \mathsf{ad} : \mathcal{C} \longrightarrow \mathcal{C}[x:X;E]$ is generic, we often elide the structure and refer to an abstraction as $\mathcal{C}[x:X;E]$.

Theorem 3.4 *The category $\mathsf{Abs}_{\mathcal{C}}$ of monoidal abstractions is equivalent with the category $\mathcal{C}_{\times}$ of commutative comonoids in $\mathcal{C}$. Each abstraction is isomorphic with the Kleisli adjunction for the comonad induced by the corresponding comonoid.*

Proof (sketch). Given a commutative comonoid $(X, \Delta, \top)$ in $\mathcal{C}$, we construct the abstraction $\mathsf{ab} \dashv \mathsf{ad} : \mathcal{C} \longrightarrow \mathcal{C}[x:X;E]$ as follows. Let

$$E \quad = \quad E_{(\Delta,\top)}$$

be the set of equations

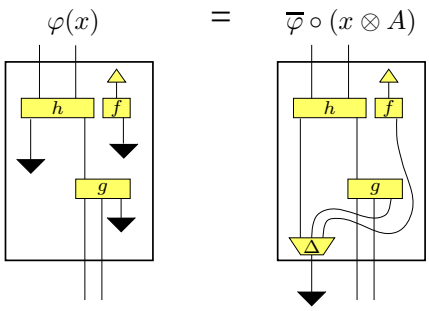
$$\underbrace{x \otimes x \otimes \cdots \otimes x}_{n \text{ times}} \quad = \quad \Delta^n \circ x \qquad \text{for } n = 0, 1, 2 \ldots$$

where $\Delta^n : X \longrightarrow X^{\otimes n}$ is defined inductively:

$$\Delta^0 = \top \qquad \Delta^1 = \mathrm{id}_X \qquad \Delta^2 = \Delta$$
$$\Delta^{i+1} = (\Delta \times X^{\otimes i-1}) \circ \Delta^i$$

This determines the extension $\mathsf{ad} : \mathcal{C} \longrightarrow \mathcal{C}[x:X;E]$. Using the symmetry, it follows that every polynomial $\varphi(x) \in \mathcal{C}[x:X;E]$ must satisfy the equation

[4] E.g., $\mathsf{Rel}[x]$ is not a locally small category.



Setting $\kappa x.\varphi(x) = \overline{\varphi}$, define

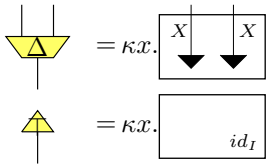
$$\begin{array}{rcl} \mathsf{ab} : \mathcal{C}[x{:}X; E] & \longrightarrow & \mathcal{C} \\ A & \longmapsto & X \otimes A \\ \varphi(x) & \longmapsto & (X \otimes \kappa x.\varphi(x)) \circ (\Delta \otimes A) \end{array} \tag{6}$$

The adjunction correspondence, with $\mathsf{ad}(B) = B$, is now

$$\mathcal{C}\big(\mathsf{ab}(A), B\big) \underset{\kappa x.}{\overset{(-)\circ(x\otimes A)}{\cong}} \mathcal{C}[x{:}X; E]\big(A, \mathsf{ad}(B)\big)$$

$$\begin{array}{ll} \big(\kappa x.\ \varphi(x)\big) \circ \big(x \otimes A\big) = \varphi(x) & (\beta\text{-rule} \\ \kappa x.\ \big(f \circ (x \otimes A)\big) = f & (\eta\text{-rule} \end{array}$$

The other way around, given an abstraction $\mathsf{ab} \dashv \mathsf{ad} : \mathcal{C} \longrightarrow \mathcal{C}[x{:}X; E]$, the conditions from Def. 3.3 imply that $h(A) = x \otimes A$ and $\mathsf{ab}(A) = X \otimes A$. With the transposition κx as above, the comonoid structure must be



The arrow part of the claimed equivalence $\mathsf{Abs}_{\mathcal{C}} \simeq \mathcal{C}_{\times}$ follows in one direction from the fact that any comonoid homomorphism $f : Y \to X$ induces a unique ioof $F : \mathcal{C}[x{:}X] \longrightarrow \mathcal{C}[y{:}Y]$, mapping $\varphi(x)$ to $F\varphi(x) = \varphi(f \circ y)$. Since every structure-preserving functor F is easily seen to be induced by the comonoid homomorphism $f = \kappa y.\ Fx$ in this way, the bijective correspondence $\mathsf{Abs}\ (\mathcal{C}[x{:}X], \mathcal{C}[y{:}Y]) \cong \mathcal{C}_{\times}(X, Y)$ is established.

The isomorphism $\mathcal{C}[x : X] \cong \mathcal{C}_{[X]}$, where $\mathcal{C}_{[X]}$ is the Kleisli category for the comonoid X, is obtained by viewing the transpositions $\kappa x.(-)$ and $(-) \circ (x \otimes A)$ as functors. More precisely, this isomorphism is realized by the following ioofs:

$$\begin{array}{rcl} K : \mathcal{C}[x{:}X] \longrightarrow \mathcal{C}_{[X]} & \qquad & H : \mathcal{C}_{[X]} \longrightarrow \mathcal{C}[x{:}X] \\ \varphi(x) \longmapsto \kappa x.\ \varphi(x) & & f \longmapsto f \circ (x \otimes A) \end{array}$$

The fact that $H \circ K = \mathrm{id}$ is just the β-rule; the fact that $K \circ H = \mathrm{id}$ is the η-rule. Proving the functoriality of K and H, and the fact that they commute with the abstraction structure $\mathsf{ab} \dashv \mathsf{ad} : \mathcal{C} \longrightarrow \mathcal{C}[x : X; E]$ and the Kleisli adjunction $V \dashv G : \mathcal{C} \longrightarrow \mathcal{C}_{[X]}$ is an instructive exercise. □

Remarks. (a) The upshot of the preceding theorem is that the set of equations E in $\mathcal{C}[x{:}X; E]$ determines the comonoid structure $(\Delta, \top)$ over X; and *vice versa*: the comonoid structure $(\Delta, \top)$ determines the equations $E = E_{(\Delta,\top)}$, as in the above proof. Just like we often speak of a "comonoid X" and leave the actual structure $(\Delta, \top)$ implicit, we shall often elide E, and write $\mathcal{C}[x{:}X]$, or even $\mathcal{C}[x]$, whenever the rest of the structure is clear from the context. We shall also blur the distinction between the comonoid $(X, \Delta, \top)$ and the corresponding comonad, and denote both by X, writing $\mathcal{C}_{[X]}$ for the X-Kleisli category, the $\mathcal{C}^{[X]}$ for the X-Eilenberg-Moore category.

(b) The extension process can be iterated to construct $\mathcal{C}[x{:}X, y{:}Y] = \mathcal{C}[x{:}X][y{:}Y] \cong \mathcal{C}_{[X\otimes Y]}$, or $\mathcal{C}[x, y{:}X \cong \mathcal{C}_{[X\otimes X]}$.

(c) The category $\mathcal{C}_\times$ of commutative comonoids is the cofree cartesian category over the monoidal category $\mathcal{C}$ [13]. The equivalence of categories established in 3.4 can be extended to an equivalence of 2-categories. The 2-cells of $\mathsf{Abs}_\mathcal{C}$ are the monoidal natural transformations. The 2-cells of $\mathcal{C}_\times$ can be obtained by dualizing the notion of natural transformations between the monoid homomorphisms. And the monoid homomorphisms are functors between categories with one object, so the usual notion of natural transformation just needs to be internalized. The reader may find it interesting to work this out.

(d) Recall (or see 2.1.2) that the tensor unit I carries a canonical structure of a commutative comonoid. Adjoining a variable $I \xrightarrow{y} I$ leads to $\mathcal{C}[y{:}I] \cong \mathcal{C}$, because $\vartriangle y = y \otimes y$ and the coherence conditions imply $y = \mathrm{id}_I$.

Corollary 3.5 *In every extension $\mathcal{C}[x{:}X]$ that supports monoidal abstraction holds $\vartriangle x = x \otimes x$ and $\top x = \mathrm{id}_I$.*

Proof. The first equation follows by postcomposing with x the equation $\vartriangle = \kappa x.\ x \otimes x$, which is the definition of $\vartriangle$ in $\mathcal{C}[x{:}X]$, and applying the β-rule. The second one is obtained by precomposing $\top = \kappa x.\ \mathrm{id}_I$ with x and applyng the β-rule. □

Corollary 3.6 *If the extension $\mathcal{C}[x{:}X]$ supports abstraction, then X is generated by the tensor unit I. As a consequence, a weakly entangled vector $\eta \in \mathcal{C}[x{:}X](X \otimes X)$ is always strongly entangled.*

Proof. By definition, I generates X in $\mathcal{C}$ if whenever $fa = ga$ for all $a \in \mathcal{C}(X)$, then $f = g$, for any $f, g \in \mathcal{C}(X, Y)$. But the η-rule implies that $fx = gx$ implies $f = g$. Hence the first claim. Furthermore, the same fact can be used to show that condition (a) implies condition (c) in Prop. 2.6. E.g., going back to the proof of 2.6, condition (c) can be obtained by composing the diagram for condition (a) and its dagger, after instantiating a to x. Condition (c) then follows by abstracting over x. □

3.2.1 Substitutions

But what does the variable x in the extension $\mathcal{C}[x]$ actually represent? What kind of vectors can be *substituted* for it?

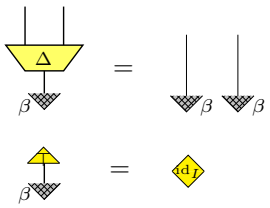
Definition 3.7 *A* Substitution *for x in $\mathcal{C}[x{:}X]$ is a monoidal functor $\mathcal{C}[x{:}X] \longrightarrow \mathcal{C}$.*

Corollary 3.8 *Substitutions $\mathcal{C}[x{:}X] \longrightarrow \mathcal{C}$ are in one-to-one correspondence with the comonoid homomorphisms $I \to X$, where X is the comonoid that induces the abstraction in $\mathcal{C}[x{:}X]$ as in Thm. 3.4.*

Remark. Only the vectors $a \in \mathcal{C}(X)$ that happen to be comonoid homomorphisms can thus be substituted for $x \in \mathcal{C}[x{:}X](X)$, leading to. In the category FHilb of finitely-dimensional Hilbert spaces, such vectors turn out to form a basis of the space X.

3.2.2 Bases

Definition 3.9 *A* basis vector *with respect to a comonoid $(X, \vartriangle, \top)$ in $\mathcal{C}$ is a comonoid homomorphism from I, i.e. an arrow $\beta : I \to X$ satisfying $\vartriangle\beta = \beta \otimes \beta$ and $\top\beta = \mathrm{id}_I$.*



The basis *of a comonoid is the set of its basis vectors.*

In Hopf algebra theory, our basis vectors are sometimes called *set-like elements*. We shall see in the next section that, for a special family of comonoids that we call classical structures, the bases tend to form categories equivalent to the category

of sets. The basis vectors of a type X in a monoidal category $\mathcal{C}$ are just the data that can be copied and deleted by a given comonoid structure on X.

Examples. Consider the monoidal category $(\mathsf{Rel}, \times, 1)$ of sets and relations. Every set X has a standard comonoid structure $X_1 = (X, \vartriangle, \top)$, induced by the cartesian structure of sets:

$$\vartriangle(x) = \{xx\} \qquad\qquad \top(x) = \{0\}$$

On the other hand, any monoid $(X, +, o)$ over the same underlying set induces a nonstandard comonoid $X_2 = (X, \widetilde{+}, \widetilde{o})$, where $\widetilde{r} : B \to A$ denotes the converse relation of $r : A \to B$, and thus

$$\widetilde{+}(u) = \{vw \mid u = v + w\} \qquad\qquad \widetilde{o}(u) = \{o\}$$

These different comonoids induce different monoidal extensions $\mathsf{Rel}[x{:}X; E_1]$ and $\mathsf{Rel}[x{:}X; E_2]$, with different abstraction operations. Both extensions have the same objects, and even the same arrows, but these arrows compose in different ways. Viewed in the Kleisli form, both categories consist of relations in the form $X \times A \longrightarrow B$. But the composites $X \times A \xrightarrow{r;s} C$ of $X \times A \xrightarrow{r} B$ and $X \times B \xrightarrow{s}$ will respectively be

$$\begin{array}{lcl} (r;s)_1(u,a,c) & \iff & \exists b.\ r(u,a,b) \wedge s(u,b,c) \\ (r;s)_2(u,a,c) & \iff & \exists bvw.\ r(w,a,b) \wedge s(v,b,c) \\ & & \qquad \wedge\ u = v + w \end{array}$$

As a consequence, each case allows substitution of different basis vectors. With respect to the standard comonoid $X_1 = (X, \vartriangle, \top)$, the basis vectors are just the singleton relations $\{u\} \in \mathsf{Rel}(X)$. The variable x in $\mathsf{Rel}[x{:}X; E_1]$ thus denotes an indeterminate element of the set X. On the other hand, with respect to the comonoid $X_2 = (X, \widetilde{+}, \widetilde{o})$, there is only one basis vector $\beta \in \mathsf{Rel}(X)$, which is the subset of X consisting of the invertible elements with respect to the monoid $(X, +, o)$. The variable x in $\mathsf{Rel}[x{:}X; E_2]$ thus denotes this one vector $\beta \in \mathsf{Rel}(X)$, since there is nothing else that can be substituted for x.

4 Daggers and classical structures

This section adds the dagger functor, and the dualities to the monoidal framework of abstraction (cf. 2.2.2). The abstraction now leads to classical structures, which were introduced in [7][5] as *classical structures.*

4.1 Dagger-monoidal abstraction

Definition 4.1 *Let $\mathcal{C}$ be a dagger-monoidal category, and E a set of equations between some parallel arrows in the dagger-monoidal polynomial category $\mathcal{C}[x{:}X]$. A* dagger-monoidal extension *is the dagger-monoidal category $\mathcal{C}[x{:}X; E] = \mathcal{C}[x{:}X]/E$, obtained by imposing the equations E on $\mathcal{C}[x{:}X]$, together with all equations that make it into a dagger-monoidal category. As all such constructions, it comes with the obvious ioof* $\mathsf{ad} : \mathcal{C} \longrightarrow \mathcal{C}[x{:}X; E]$.

A substitution functor *between the dagger-monoidal extensions is a monoidal ioof $F : \mathcal{C}[x{:}X; E] \longrightarrow \mathcal{C}[y{:}Y; D]$ which preserves the dagger, i.e. $F(\psi^\ddagger) = (F\psi)^\ddagger$.*

We denote by $\ddagger$-$\mathsf{Ext}_\mathcal{C}$ the category of dagger-monoidal extensions of $\mathcal{C}$, with the substitution functors between them.

Definition 4.2 *A* dagger monoidal abstraction *over a dagger monoidal extension $\mathsf{ad} : \mathcal{C} \longrightarrow \mathcal{C}[x{:}X; E]$ is the adjunction $\mathsf{ab} \dashv \mathsf{ad}$, which satisfies the requirements of Definition 3.3, and moreover preserves the dagger, in the sense that $\mathsf{ab}.\varphi(x)^\ddagger = (\mathsf{ab}.\varphi(x))^\ddagger$.*

We denote by $\ddagger$-$\mathsf{Abs}_\mathcal{C}$ the subcategory of $\ddagger$-$\mathsf{Ext}_\mathcal{C}$ where the abstraction is supported. Its objects are often called abstractions.

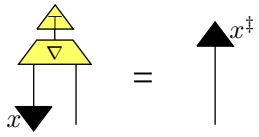
[5] Their origin in the abstraction operations was not addressed there.

Thm. 3.4 established the correspondence between monoidal abstractions over X and the comonoid structures carried by X. The next theorem extends this correspondence to dagger monoidal categories: a monoidal abstraction corresponding to a comonoid structure preserves the dagger if and only if the Kleisli category, induced by the comonoid, is (equivalent with) the dagger monoidal extension itself.

Theorem 4.3 *Let $\mathcal{C}$ be a dagger-monoidal category and* $\mathsf{ad} : \mathcal{C} \longrightarrow \mathcal{C}[x{:}X; E]$ *a dagger-monoidal extension. Suppose that it admits a monoidal abstraction* $\mathsf{ab} \dashv \mathsf{ad}$ *(as in Def. 3.3), with the induced comonoid $(X, \vartriangle, \top)$ (as in Thm. 3.4) . Then the following statements are equivalent:*

(a) $\mathsf{ab} \dashv \mathsf{ad} : \mathcal{C} \longrightarrow \mathcal{C}[x{:}X; E]$ *is a dagger-abstraction, i.e.* $\mathsf{ab}.\varphi(x)^\ddagger = (\mathsf{ab}.\varphi(x))^\ddagger$

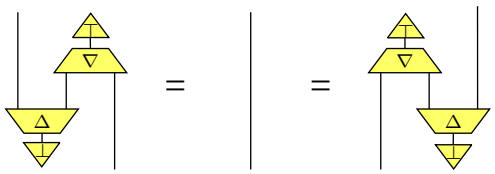
(b) x is real, i.e. $x^* = x^\ddagger$



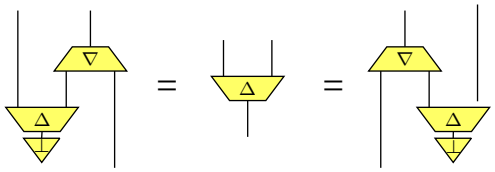
(c) $\mathsf{ab} \dashv \mathsf{ad} : \mathcal{C} \longrightarrow \mathcal{C}[x{:}X; E]$ *is isomorphic with the Kleisli adjunction $V \dashv G : \mathcal{C} \longrightarrow \mathcal{C}_{[X]}$*

The following conditions provide further equivalent characterizations of (a-c), this time expressed in terms of the properties of the comonoid $(X, \vartriangle, \top)$ and its dual monoid $(X, \triangledown, unt)$, where $\triangledown = \vartriangle^\ddagger$ and $\bot = \top^\ddagger$.

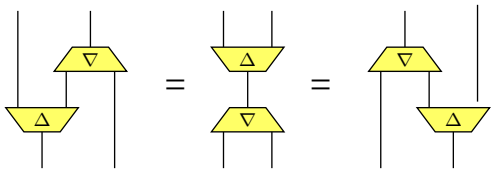
(i) $\eta = \vartriangle \circ \bot$ *and* $\varepsilon = \top \circ \triangledown$ *make $X = X^*$ self-dual*



(ii) $(X \otimes \triangledown) \circ (\eta \otimes X) = \vartriangle = (\triangledown \otimes X) \circ (X \otimes \eta)$

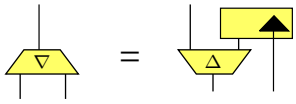


(iii) $(X \otimes \triangledown) \circ (\vartriangle \otimes X) = \vartriangle \circ \triangledown = (\triangledown \otimes X) \circ (X \otimes \vartriangle)$



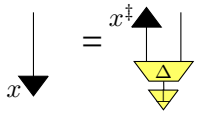
Remark. Condition (iii) is the *Frobenius condition*, analyzed in [5, 4, 20, 7]. Condition (ii) is Lawvere's earlier version of the same [25]. In each of the last three conditions, the commutativity assumption makes one of the equations redundant. The equivalence of (i-iii), however, holds without this commutativity.

Proof. **(a⇒b)** Using the definition (6) of ab, condition (a) implies that $\nabla = (\mathsf{ab}.x)^\ddagger = \mathsf{ab}.x^\ddagger = (X \otimes \kappa x.x^\ddagger) \circ (\Delta \otimes X)$, or graphically

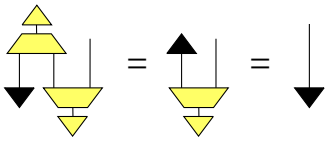


from which (b) follows by precomposing both sides with $(x \otimes X)$ and postcomposing with $\top$.

(b⇒i) Dualizing (b) gives $x = x_* = x^{\ddagger *}$, i.e.

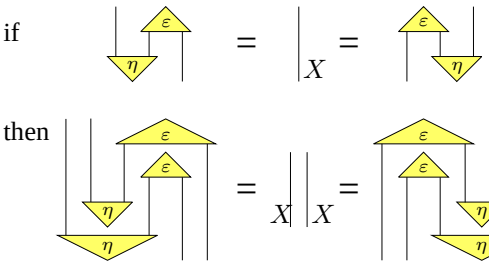


Combining (b) and its dual gives

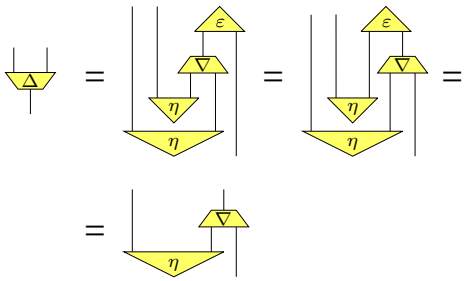


from which (i) follows, because the η-rule implies that $f \circ (x \otimes A) = g \circ (x \otimes A) \Longrightarrow f = g$

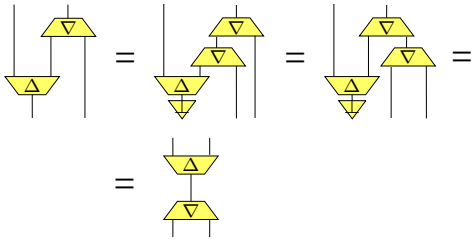
(i⇒ii) On one hand, if X is self-dual, then $X \otimes X$ is self-dual too, because



On the other hand, (i) also implies that $\nabla^\ddagger = \nabla^*$, and since $\Delta = \nabla^\ddagger$ holds by definition, we have

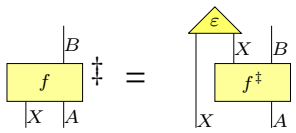


(ii⇒iii) Using (ii) to expand Δ at the first step, and to collapse it at the last step, we get



(iii⇒i) follows in a way obvious from the diagrams, by precomposing the first equation of (iii) with $\bot \otimes X$ and postcomposing it with $X \otimes \top$; and by precomposing the second equation with $X \otimes \bot$ and postcomposing it with $\top \otimes X$.

(i⇒c) Using the self-duality of X, the dagger on $\mathcal{C}_{[X]}$ is defined by



Since this implies $\kappa x.\ \varphi(x)^\ddagger = (\kappa x.\ \varphi(x))^\ddagger$, it follows that the isomorphism $\mathcal{C}[x : X] \cong \mathcal{C}_{[X]}$, defined in the proof of Thm. 3.4, preserves the dagger.

(c⇒a) Since the dagger preservation under the isomorphism $\mathcal{C}[x{:}X] \cong \mathcal{C}_{[X]}$ means that the dagger in $\mathcal{C}_{[X]}$ must be as above, it follows

By (6), the left-hand side is $\mathsf{ab}.\ \varphi(x)^\ddagger$, whereas the right-hand side is $(\mathsf{ab}.\ \varphi(x))^\ddagger$. Hence (a). □

Definition 4.4 *A* Frobenius algebra *in a monoidal category $\mathcal{C}$ is a structure $(X, \nabla, \Delta, \bot, \top)$ such that*

- *$(X, \nabla, \bot)$ is a monoid,*
- *$(X, \Delta, \top)$ is a comonoid, and*
- *the equivalent conditions (i-iii) of Thm. 4.3 are satisfied.*

A dagger-Frobenius algebra *in a dagger-monoidal category $\mathcal{C}$ is a Frobenius algebra where $\nabla = \Delta^\ddagger$ and $\bot = \top^\ddagger$.*

Thm. 4.3 can now be summarized as follows.

Corollary 4.5 *The category of dagger-monoidal abstractions $\ddagger$-$\mathsf{Abs}_\mathcal{C}$ is equivalent with the category $\mathcal{C}_\Delta$ of commutative dagger-Frobenius algebras and comonoid homomorphisms in $\mathcal{C}$.*

Summary. The upshot of Thm. 4.3 is thus that a monoidal extension $\mathcal{C}[x : X]$, induced by a commutative comonoid X which also happens to be a dagger-Frobenius algebra, is necessarily a dagger-monoidal extension. The immediate corollary is the following.

Corollary 4.6 *The substitutions $\mathcal{C}[x{:}X] \longrightarrow \mathcal{C}$ of the basis vectors with respect to a Frobenius algebra X preserve not only the tensors and their unit, but also the daggers.*

Furthermore, since the basis vectors of the Frobenius algebra X are substituted for the variable x, which must be real, it is natural to expect, and easy to prove that

Corollary 4.7 *The basis vectors with respect to a dagger-Frobenius algebra are always real.*

Remark. This last statement may sound curious. There are many complex vectors in a complex Hilbert space, and each of them may participate some basis. However, after a change of basis they may become real; and some vectors that were real will cease to be real. The notion of reality depends on the choice of basis. However, just like people, the basis vectors themselves

always satisfy their own notion of reality: they are in the form $\beta_1 = (1,0,0,\ldots,0), \beta_2 = (0,1,0,\ldots,0), \ldots, \beta_n = (0,0,\ldots,0,1)$.

4.2 Classical structures

It turns out that Frobenius algebras with additional properties provide a purely algebraic characterization of the choice of a basis, e.g. in a Hilbert space. More generally, in an abstract quantum universe, we can thus distinguish classical data types, by means of algebraic operations. We begin by describing the additional property needed for this.

Lemma 4.8 *Let $\mathcal{C}[x, y : X]$ be a dagger-monoidal extension induced by the Frobenius algebra $(X, \nabla, \Delta, \bot, \top)$. Then the following conditions are equivalent:*

(a) $\nabla \circ \Delta = \mathrm{id}_X$

(b) $\nabla(x \otimes x) = x$

(c) $\langle x|y\rangle^2 = \langle x|y\rangle$

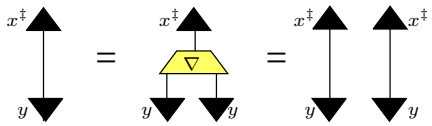
and they imply

(d) $\langle x|x\rangle = \mathrm{id}_I$

The equivalence of (a) and (b) is also valid for monoidal categories, with no dagger.

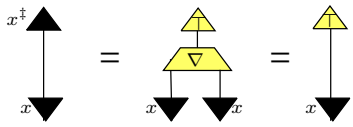
Proof. **(a⇒b)** $\nabla(x \otimes x) = \nabla\Delta x = x$, using Cor. 3.5.

(b⇒c) $\langle x|y\rangle = x^\ddagger \circ y = x^\ddagger \circ \nabla \circ (y \otimes y) = x^\ddagger \circ \Delta^\ddagger \circ (y \otimes y) = (x^\ddagger \otimes x^\ddagger)(y \otimes y) = (x^\ddagger \circ y) \otimes (x^\ddagger \circ y) = \langle x|y\rangle^2$, i.e.



(c⇒a) $x^\ddagger \circ \nabla \circ \Delta \circ y = (x^\ddagger \otimes x^\ddagger)(y \otimes y) = (x^\ddagger \circ y) \otimes (x^\ddagger \circ y) = \langle x|y\rangle^2 = \langle x|y\rangle = x^\ddagger \circ y$, and then use the η-rule.

(b⇒d) Since by Thm. 4.3 $x^\ddagger = x^*$, and by Cor. 3.5 $\top x = \mathrm{id}_I$, we have $\langle x|x\rangle = x^\ddagger x = \top x = \mathrm{id}_I$.



□

Definition 4.9 *A* classical structure *is a commutative dagger-Frobenius algebra satifsying 4.8(a). A* classical extension *of $\mathcal{C}$ is a dagger-monoidal extension $\mathcal{C}[x:X]$ induced by a classical structure, i.e. satisfying 4.8(b-c).*

Remark. Lemma 4.8(b) and Thm. 4.3 together say that a monoidal extension $\mathcal{C}[x:X]$ of a dagger monoidal category $\mathcal{C}$ is a classical extension if and only the variable x is real and idempotent, i.e. $x = x_* = x \bullet x$, where $a \bullet b = \nabla(a \otimes b)$ is the convolution, mentioned in 2.1.5. Lemma 4.8(c) says that the idempotence of x is equivalent with the idempotence of the inner product $\langle x|y\rangle$ of any two variables of type X. (Idempotence with respect to which monoid? Recall from Sec. 2.1.5 that the convolution, the composition, and the tensor of scalars all induce the same monoid, since $s \bullet t = s \circ t = s \otimes t$ holds for all $s, t \in \mathcal{C}(I)$.)

Note that, by the η-rule, $\langle x|y\rangle = \langle x|z\rangle \Rightarrow y = z$. It follows that the monoid of scalars in a polynomial extension $\mathcal{C}[x, y, z:X]$ must have freshly adjoined elements, if $x \neq y \neq z$. Another interesting point is that the implication $\langle x|a\rangle = \langle x|b\rangle \Rightarrow a = b$,

valid in $\mathcal{C}[x:X]$, is preserved under the substitutions *jointly*, provided that the basis vectors generate X: if $\langle\beta|a\rangle = \langle\beta|b\rangle$ holds for all basis vectors β, then $a = b$. Elaborating this, one could formulate the suitable soundness and completeness notions and for reasoning with polynomials and classical structures, but we shall not pursue this thread.

Corollary 4.10 *The category $\mathcal{B}$-Abs$_\mathcal{C}$ $\subseteq$ ‡-Abs$_\mathcal{C}$ of classical abstractions of $\mathcal{C}$ is equivalent with the category $\mathcal{C}_\mathcal{B}$ of classical structures and comonoid homomorphisms in $\mathcal{C}$.*

Note that the category $\mathcal{C}_\mathcal{B}$ is a cartesian subcategory of the category $\mathcal{C}_\times$ of commutative comonoids. While the forgetful functor $\mathcal{C}_\times \longrightarrow \mathcal{C}$ was couniversal for all monoidal functors from cartesian categories to $\mathcal{C}$, the forgetful functor $\mathcal{C}_\mathcal{B} \longrightarrow \mathcal{C}$ is couniversal for the conservative functors among them. The exactness properties of $\mathcal{C}_\mathcal{B}$, induced by the various properties of $\mathcal{C}$, were analyzed in [4]. If $\mathcal{C}$ is compact [18] and right exact with biproducts, then $\mathcal{C}_\mathcal{B}$ turns out to be a pretopos. In any case, if $\mathcal{C}$ represents a quantum universe, $\mathcal{C}_\mathcal{B}$ can be thought of as the category of classical data.

4.2.1 Orthonormality of bases

Definition 3.9 stipulated an abstract notion of a basis with respect to a comonoid. The notion of a classical structure now characterizes just those comonoids whose bases are *orthonormal*, in the sense of the following

Definition 4.11 *A vector $a \in \mathcal{C}(A)$ is* normalized *if $\langle a|b\rangle = \mathrm{id}_I$. A pair of vectors $a, b \in \mathcal{C}(A)$ is* orthogonal *if $\langle a|b\rangle^2 = \langle a|b\rangle$. A set of vectors is* orthonormal *when each element is normalized, and each pair orthogonal.*

Lemma 4.8 and Cor. 3.8 imply that

Proposition 4.12 *The basis set of every classical structure is orthonormal.*

4.2.2 Succinct classical structures

The following lemma shows that being a classical structure is a property of a comonoid (or of a monoid), rather than additional structure.

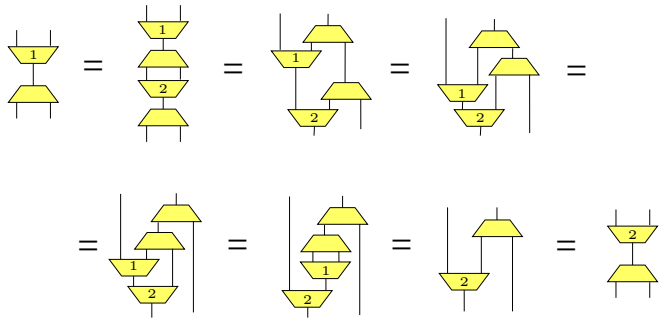
Lemma 4.13 *The monoid and the comonoid part of a classical structure determine each other: e.g., $(X, \nabla, \vartriangle_1, \bot, \top_1)$ and $(X, \nabla, \vartriangle_2, \bot, \top_2)$ are classical structures, then $\vartriangle_1 = \vartriangle_2$ and $\top_1 = \top_2$.*

Since $(X, \nabla, \vartriangle, \bot, \top)$ is completely determined by $(X, \nabla, \bot)$ (and by $(X, \vartriangle, \top)$), it is justified to speak succinctly of the classical structure $(X, \nabla, \bot)$ (and of the classical structure $(X, \vartriangle, \top)$).

Proof. It is enough to prove $\vartriangle_1 \circ \nabla = \vartriangle_2 \circ \nabla$, because this and $\nabla \circ \vartriangle_1 = \mathrm{id}_X$ give

$$\vartriangle_1 \;=\; \vartriangle_1 \circ \nabla \circ \vartriangle_1 \;=\; \vartriangle_2 \circ \nabla \circ \vartriangle_1 \;=\; \vartriangle_2$$

Here is a diagrammatic proof $\vartriangle_1 \circ \nabla = \vartriangle_2 \circ \nabla$:



□

4.2.3 Classifying classical structures

Proposition 4.14 [10] *In the category* $(\mathsf{FHilb}, \otimes, \mathbb{C}, \ddagger)$ *of finitely-dimensional complex Hilbert spaces and linear maps, the classical structures correspond to the orthonormal bases in the usual sense.* $\mathsf{FHilb}_{\mathcal{B}}$ *is equivalent with the category* FSet *of finite sets and functions.*

Proposition 4.15 [31] *In the category* $\left(\mathsf{Rel}, \times, 1, \widetilde{(-)}\right)$ *of sets and relations, the classical structures are just the biproducts (disjoint unions) of abelian groups.* $\mathsf{Rel}_{\mathcal{B}}$ *is equivalent with the category* Set *of sets and functions.*

Each classical structure X in Rel decomposes as a disjoint union $X = \sum_{j\in J} X_j$ where each restriction $(X_j, \triangledown_j, \perp_j)$ of $(X, \triangledown, \perp)$ is an abelian group. A classical structure on X thus consists of (1) a partition $X = \sum_{j\in J} X_j$ and (2) an abelian group structure on each X_j. These partitions and group structures, and even the size of X are, however, indistinguishable by the morphisms of $\mathsf{Rel}_{\mathcal{B}}$, because any two classical structures with the same number J of components are isomorphic.

Bases in Rel**.** The basis induced by the classical structure $X = \sum_{j\in J} X_j$ is in the form $\mathcal{B}(X) = \{X_j\}_{j\in J}$. While the bases with the same number of elements are indistinguishable in $\mathsf{Rel}_{\mathcal{B}}$, they are the crucial resource for quantum computation in Rel. The bases induced by the *rectangular* structures $(\Xi_n, \vartriangle, \top)$, will be particularly useful, where

$$\begin{aligned}
\Xi_n &= \sum_n \mathbb{Z}_n = \{ij \mid 0 \leq i, j \leq n-1\} \\
\vartriangle(ij) &= \{\langle ik, i\ell\rangle \mid j = k + \ell\} \\
\top &= \{i0 \mid 0 \leq i \leq n-1\} \\
\mathcal{B}(\Xi_n) &= \{\beta_i = \{ij\} \mid 0 \leq i, j \leq n-1\}
\end{aligned}$$

4.3 Bases for Simon's algorithm

Any bitstring function $f : \mathbb{Z}_2^m \to \mathbb{Z}_2^n$, considered in Simon's algorithm, can be viewed as a morphism $f \in \mathsf{FSet}_{\wp}(m, n)$ in the category of finite powersets and all functions between them. It is easy to see that this is a cartesian closed category, with $+$ as the cartesian product[6]. The program transformation from the function f to the corresponding Hilbert space unitary U_f is formalized as follows

$$\frac{f(x) = f \circ x \in \mathsf{FSet}_{\wp}[x{:}m](n)}{\dfrac{f'(x,y) = \langle x, y \oplus f(x)\rangle \in \mathsf{FSet}_{\wp}[x, y{:}m+n](m+n)}{U_f|x,y\rangle = \mathbb{B}^{\otimes f'(x,y)} \in \mathsf{FHilb}\left[|x,y\rangle{:}\mathbb{B}^{\otimes(m+n)}\right]\left(\mathbb{B}^{\otimes(m+n)}\right)}}$$

where $\mathbb{B} = \mathbb{C}^2$. The unitary U_f is thus the image of f' along the functor

$$\mathbb{B}^{\otimes(-)} \;:\; \mathsf{FSet}_{\wp}[x, y{:}m+n] \longrightarrow \mathsf{FHilb}\left[|x,y\rangle{:}\mathbb{B}^{\otimes(m+n)}\right]$$

which maps finite sets to the tensor powers of $\mathbb{B}$. Since $\mathbb{B}^{\otimes m} = \mathbb{C}^{(2^m)}$, any function $f : 2^m \to 2^n$ in $\mathsf{Set}_{\wp}$ is mapped to a linear operator $\mathbb{B}^{\otimes f} : \mathbb{B}^{\otimes m} \longrightarrow \mathbb{B}^{\otimes n}$ in FHilb, represented by the matrix $F = (F_{ij})_{2^n \times 2^m}$ where $F_{ij} = 1$ whenever $f(j) = i$, otherwise $F_{ij} = 0$. This determines a functor $\mathsf{FSet}_{\wp} \longrightarrow \mathsf{FHilb}$. It is extended to a substitution $\mathsf{FSet}_{\wp}\,[x, y{:}m+n] \longrightarrow \mathsf{FHilb}\left[|x,y\rangle{:}\mathbb{B}^{\otimes(m+n)}\right]$ by stipulating that the variables x, y are mapped to the variables $|x, y\rangle$.

The function $f \in \mathsf{FSet}_{\wp}(m, n)$ has a simpler, though nonstandard interpretation in the dagger-*pre*monoidal[7] category $(\mathsf{Rel}_{\wp}, \otimes, 1, \ddagger)$, where $\mathsf{Rel}_{\wp}(m, n) = \mathsf{Rel}(2^m, 2^n)$ and $m \otimes n = m \times n$. The dagger is still just the relational converse. Like before, we define

$$\Xi^{\otimes(-)} \;:\; \mathsf{FSet}_{\wp}[x, y{:}m+n] \longrightarrow \mathsf{Rel}_{\wp}\left[|x,y\rangle{:}\Xi^{\otimes(m+n)}\right]$$

[6] $\mathsf{FSet}_{\wp}$ is opposite to the Kleisli category for the $\wp\wp$-monad. Along the discrete Stone duality, $\mathsf{FSet}_{\wp}$ is thus dual to the category of free finite atomic Boolean algebras. Since Boolean algebras are primal, every function between them can be expressed as a polynomial.

[7] The tensor $m \otimes n = m \times n$ is functorial in each argument, but it is not a bifunctor. See [34] for a discussion about such structures. This has no repercussions for us, since the definition of the functor $\Xi^{\otimes(-)}$, spelled out explicitly below, makes no use of the arrow part of $\otimes$.

this time over the rectangular structure

$$\Xi = \Xi_2 = \{00, 01, 10, 11\}$$
$$\vartriangle(i0) = \{\langle i0, i0\rangle, \langle i1, i1\rangle\} \quad \vartriangle(i1) = \{\langle i0, i1\rangle, \langle i1, i0\rangle\}$$
$$\top = \{00, 10\}$$
$$\mathcal{B}(\Xi) = \{\beta_0 = \{00, 01\}, \beta_1 = \{10, 11\}\}$$

Note that this comonoid structure lifts from $(\mathsf{Rel}, \times, 1)$ to $(\mathsf{Rel}_\wp, \otimes, 1)$ because $\Xi \otimes \Xi = 2^2 \otimes 2^2 = 2^{2\times 2} = 2^{2+2} = 2^2 \times 2^2 = \Xi \times \Xi$. It furthermore lifts to any $\Xi^{\otimes m}$, since the commutative (co)monoid structures always extend to the tensor powers.

Since the underlying set of $\Xi^{\otimes m}$ is $2^{(2^m)}$, any function $f : 2^m \to 2^n$ in $\mathsf{Set}_\wp$, is mapped to a relation $\Xi^{\otimes f} : \Xi^{\otimes m} \longrightarrow \Xi^{\otimes n}$ in $\mathsf{Rel}_\wp$, represented by the matrix $F = (F_{ij})_{2^n \times 2^m}$ where $F_{ij} = 1$ whenever $f(j) = i$, otherwise $F_{ij} = 0$. The functor is extended into a substitution $\mathsf{Set}_\wp[x, y : m + n] \longrightarrow \mathsf{Rel}_\wp\left[|x, y\rangle : \Xi^{\otimes(m+n)}\right]$ like before. Mapping the polynomial $f'(x, y)$, constructed above, along this functor, we get a polynomial unitary relation $\Upsilon_f|x, y\rangle = \Xi^{\otimes f'(x,y)}$ on $\Xi^{\otimes(m+n)}$ in $\mathsf{Rel}_\wp\left[|x, y\rangle : \Xi^{\otimes(m+n)}\right]$. This polynomial can be viewed as a family of unitary relations indexed over the basis of $\Xi^{\otimes(m+n)}$; and each member of the family is a permutation on $\Xi^{\otimes(m+n)} = 2^{\left(2^{m+n}\right)}$.

5 Complementarity

5.1 Complementary classical structures

Definition 5.1 *A vector $a \in \mathcal{C}(X)$ is* unbiased *(or complementary) with respect to a classical structure $(X, \vartriangle, \top)$ if $\vartriangle a \in \mathcal{C}(X \otimes X)$ is strongly entangled (in the sense of Sec. 2.2.4). Two classical structures are complementary if every every basis vector with respect to one is complementary with respect to the other one, and* vice versa.

Remark. In the framework of Hilbert spaces, this definition is equivalent to the standard notion of complementary bases, used for describing the quantum uncertainty relations [21, 40]. Coecke, Duncan and Edwards [8, 9] have characterized complementary vectors in terms of their representations (cf. Sec. 2.1.5 (2)). The first part of the following proposition says that our definition is equivalent to theirs.

Proposition 5.2 *With respect to a classical structure X, the representative $\widehat{b} \in \mathcal{C}(X, X)$ of $b \in \mathcal{C}(X)$ is*

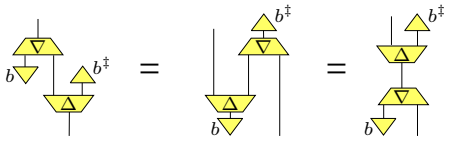
(a) unitary if and only if b is unbiased;

(b) a pure projector if b is a basis vector.

The converse of (b) holds whenever the basis vectors generate X.

Recall from Sec. 2.2.2 that the usual definitions of projectors and unitaries lift to dagger-categories: a unitary is an endomorphism u such that $u^\ddagger = u^{-1}$, whereas a projector p satisfies $p = p^\ddagger = p \circ p$. For a pure projector over X we moreover require $\mathrm{Tr}(p) = \varepsilon \circ (X \otimes p) \circ \eta = \mathrm{id}_I$. The assumption that a set of vectors $\Gamma \subseteq \mathcal{C}(X)$ generates an object X means that for any $f \neq g \in \mathcal{C}(X, Y)$ there must be a basis vector $a \in \Gamma$ such that $fa \neq ga$.

Proof of 5.2. **(a)** Since $\triangledown$ is commutative, by the definition of $\widehat{b}$ in (2), $\widehat{b^\ddagger} = (\triangledown(b \otimes X))^\ddagger = \left(X \otimes b^\ddagger\right)\vartriangle$. The composites $\widehat{b} \circ \widehat{b^\ddagger}$ and $\widehat{b^\ddagger} \circ \widehat{b}$ can thus be viewed as the left-hand side and the right-hand side of the following diagram.

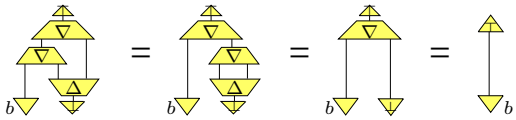


Both side diagrams can be transformed into the middle one by applying the Frobenius condition 4.3(iii). Thus

$$\widehat{b} \circ \widehat{b}^{\ddagger} = \mathrm{id}_X \quad \Longleftrightarrow \quad (X \otimes b^{\ddagger}\nabla)(\triangle b \otimes X) = \mathrm{id}_X \quad \Longleftrightarrow \quad \widehat{b}^{\ddagger} \circ \widehat{b} = \mathrm{id}_X$$

But by Defn. 2.5, the middle equation just says that $\triangle b$ is strongly entangled, i.e. that b is unbiased. Hence the claim.

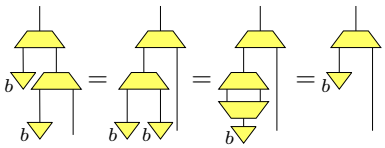
(b) To begin from the easiest, first note that $\mathrm{Tr}(\widehat{b}) = \mathrm{id}_I \iff \top b = \mathrm{id}_I$, because $\mathrm{Tr}(\widehat{b}) = \top b$:



Secondly, we want to show that $\widehat{b} = \widehat{b}^{\ddagger} \iff b^* = b^{\ddagger}$, i.e.

The right-hand equation says that b is real, which is a property of every basis vector, according Cor. 4.7. The implication from left to right is obtained by postcomposing both sides of the left-hand equation with $\top$. The implication from right to left is obtained by tensoring by X on the right both sides of the right-hand equation, and then precomposing them with $\triangle$. The left-hand equation is then obtained using 4.3(ii).

To complete the proof, we show that $\triangle b = b \otimes b$ implies $\widehat{b} \circ \widehat{b} = \widehat{b}$, by the following diagram:



□

5.2 Transforms

A given basis of a Hilbert space can be mapped into a complementary one using a Fourrier transform. This is done in all HSP-algorithms: the basis vectors are entangled into one complementary vector, and the unitary U_f is then evaluated over that vector, thus computing all values of f in one sweep.

In order to complete the implementation of Simon's algorithm in $\mathsf{Rel}_{\wp}$, we need a pair of complementary bases for $\Xi^{\otimes(m+n)}$. As mentioned above, the classical structures of Ξ lift from Rel to $\mathsf{Rel}_{\wp}$. And in Rel in general, for a given classical structure $X = \sum_{j \leq m} X^1_j$ in Rel, a complementary vector is a set $\gamma \subseteq X$ such that $\gamma_j = \gamma \cap X^1_j$ is a singleton for every $j \leq m$. Another classical structure $X = \sum_{k \leq n} X^2_k$ over the same set is thus complementary if and only if $X^1_j \cap X^2_k$ is a singleton for all $j \leq m, k \leq n$. Since X^1 and X^2 are partitions, it follows that all $\#X^1_j = n$ and all $\#X^2_k = m$. So X must decompose to m groups of order n, and to n groups of order n. In order to have an invertible transform from one basis to another, we need $m = n$. Unless we are interested in the various forms of entanglement engendered by the various group structures, we can thus restrict attention to rectangular structures from sec. 4.2.3. A simple transform mapping the basis vectors of Ξ_ℓ of into a complementary basis is

$$\begin{array}{rccc} H_\ell & : & \Xi_\ell & \longrightarrow & \Xi_\ell \\ & & ij & \longmapsto & ji \end{array}$$

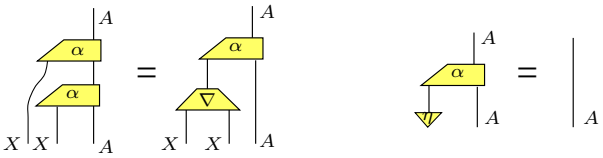
Using $H = H_2$ to transform $H^{\otimes m} : \Xi^{\otimes m} \longrightarrow \Xi^{\otimes m}$ we can now produce the superposition of all the basis vectors, representing the inputs of the function $f : \mathbb{Z}_2^m \to \mathbb{Z}_2^n$ from Simon's algorithm. The other way around, the H-image of any basis vector is the superposition of the complementary basis of $\Xi^{\otimes m}$. We can thus define the unitary polynomial $(H^{\otimes m} \otimes \mathrm{id}) \circ \Upsilon_f |x, y\rangle \circ (H^{\otimes m} \otimes \mathrm{id})$ on $\Xi^{\otimes(m+n)}$ in $\mathsf{Rel}_{\wp}\left[|x, y\rangle : \Xi^{\otimes(m+n)}\right]$ and evaluate it on the vector $|0, 0\rangle = \perp \in \mathsf{Rel}_{\wp}\left(\Xi^{\otimes(m+n)}\right)$, to get the outcome $S|x, y\rangle \in \mathsf{Rel}_{\wp}\left[|x, y\rangle : \Xi^{\otimes(m+n)}\right]\left(\Xi^{\otimes(m+n)}\right)$. To complete the execution of Simon's algorithm in $\mathsf{Rel}_{\wp}$, we just need to measure this outcome.

6 Measurements

So far, we have seen that the classical data in a quantum universe, represented by a dagger-monoidal category $\mathcal{C}$, can be characterized as just those data that can be annotated by the variables in $\mathcal{C}[x, y, \ldots]$, i.e. those data that support the abstraction operation κx. Quantum programs are thus viewed as polynomial arrows $\varphi(x, y, \ldots) \in \mathcal{C}[x, y, \ldots]$. In this respect, quantum programs are similar to classical programs: they specify that some operations should be applied to some input data, always classical, denoted by the variables. Semantics of computation is captured through abstractions and subsitutions. Program execution, in particular, corresponds to substituting some input data for the variables, and evaluating the resulting expressions.

In classical computation, such evaluations yield the outputs. In quantum computation, however, there is more: the outputs need to be *measured*. The view of quantum programs as polynomials in dagger-monoidal categories needs to be refined to capture measurements. In the simplest case, a measurement will turn out to be just a projector in $\mathcal{C}[x{:}X]$.

Definition 6.1 *A morphism $X \otimes A \xrightarrow{\alpha} A$ in $\mathcal{C}$ on is an X-*action *A if $\alpha \circ (X \otimes \alpha) = \alpha \circ \nabla$. An X-action is* normal *if moreover $\alpha \circ (\bot \times A) = \mathrm{id}_A$.*



An X-equivariang homomorphism from $X \otimes A \xrightarrow{\alpha} A$ to $X \otimes B \xrightarrow{\beta} B$ is an arrow $f \in \mathcal{C}(A, B)$ such that $f \circ \alpha = \beta \circ (X \otimes f)$. The category of X-actions and X-equivariang homomorphisms is denoted $\mathcal{C}^{\{X\}}$.

The full subcategory of normal *X-actions is $\mathcal{C}^{[X]} \hookrightarrow \mathcal{C}^{\{X\}}$.*

Remark. Normal X-actions are the Eilenberg-Moore algebras for the monad $X \otimes (-) : \mathcal{C} \longrightarrow \mathcal{C}$. Equivalently, they are also actions of the monoid X, and this terminology tends to lead to less confusion.

Lemma 6.2 *Let $(X, \Delta, \top)$ be a classical structure, $\alpha(x) : A \longrightarrow A$ an endomorphism in $\mathcal{C}[x : X]$ and $\alpha = \kappa x.\ \alpha(x) : X \otimes A \longrightarrow A$ its abstraction.*

(a) The following conditions are equivalent:

(i) $\alpha(x) = \alpha(x) \circ \alpha(x)$, *i.e. $\alpha(x)$ is idempotent*

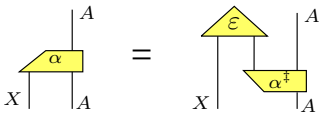
(ii) $\alpha \circ (X \otimes \alpha) = \alpha \circ \nabla$, *i.e. α is an X-action*

(iii) $\alpha \circ (X \otimes \alpha) \circ (\Delta \otimes A) = \alpha$, *i.e. α is idempotent as an endomorphism on A in $\mathcal{C}_{[X]}$.*

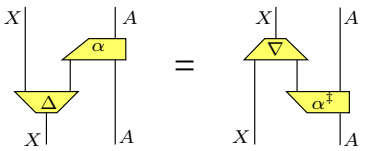
(b) On the other hand, the following conditions are also equivalent:

(i) $\alpha(x) = \alpha(x)^\ddagger$, *i.e. $\alpha(x)$ is self-adjoint*

(ii) $\alpha = (\varepsilon \otimes A) \circ \alpha^\ddagger$



(iii) $(X \otimes \alpha) \circ (\Delta \otimes A) = (\nabla \otimes A) \circ (X \otimes \alpha^\ddagger)$



The **proofs** of the above equivalences are easy exercises with classical structure. The equivalence (b)(ii⇔iii) can be viewed, and proven, in analogy with Thm. 4.3(ii⇔iii).

Definition 6.3 *Let X be a classical structure in $\mathcal{C}$. An X-*measurement *over $A \in \mathcal{C}$ is a projector $\alpha(x) : A \longrightarrow A$ in $\mathcal{C}[x{:}X]$, i.e. a self-adjoint idempotent $\alpha(x) = \alpha(x)^\ddagger = \alpha(x) \circ \alpha(x)$.*

A homomorphism $f : \alpha(x) \longrightarrow \beta(x)$, where $\alpha(x)$ is an X-measurement over A and $\beta(x)$ is an X-measurement over B, is an arrow $f \in \mathcal{C}(A, B)$ such that $f \circ \alpha(x) = \beta(x) \circ f$. The category of measurements in the classical structure $(X, \Delta, \top)$ is denoted by $\mathcal{C}\{x{:}X\}$.

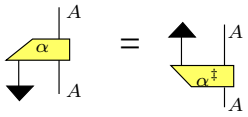
Remark. Substituting a basis vector $\varphi \in \mathcal{B}(X)$ into a measurement $\alpha(x) \in \mathcal{C}[x : X](A, A)$ yields a projector $\alpha(\varphi) \in \mathcal{C}(A, A)$. The intuition is that this projector corresponds to an the outcome of the measurement α.

It is easy to see that $\mathcal{C}\{x{:}X\}$ is a dagger-monoidal category. The following two propositions show that this notion of a measurement is equivalent with the one from [7].

Theorem 6.4 *Let X be a classical structure, and $\alpha(x) : A \longrightarrow A$ an endomorphism in $\mathcal{C}[x{:}X]$. Then (a) $\iff$ (b) $\Longleftarrow$ (c).*

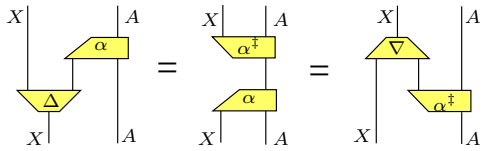
(a) $\alpha(x) : A \longrightarrow A$ *is a measurement*

(b) $\alpha = \kappa x.\, \alpha(x) : XA \longrightarrow A$ *is an X-action such that* $\alpha \circ (x \otimes A) = (x^\ddagger \otimes A) \circ \alpha^\ddagger$

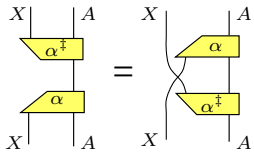


(c) α *is an X-action satisfying the following equivalent conditions*

(i) $(X \otimes \alpha) \circ (\Delta \otimes A) \;=\; \alpha^\ddagger \circ \alpha \;=\; (\nabla \otimes A) \circ (X \otimes \alpha^\ddagger)$

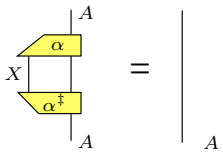


(ii) $\alpha^\ddagger \circ \alpha = (X \otimes \alpha) \circ (c \otimes A) \circ (X \otimes \alpha^\ddagger)$



The converse (c) $\Longrightarrow$ (a) $\wedge$ (b) holds if the X-action α is normal. When this is the case, then also

$$\alpha \circ \alpha^\ddagger \;=\; \mathrm{id}_A$$



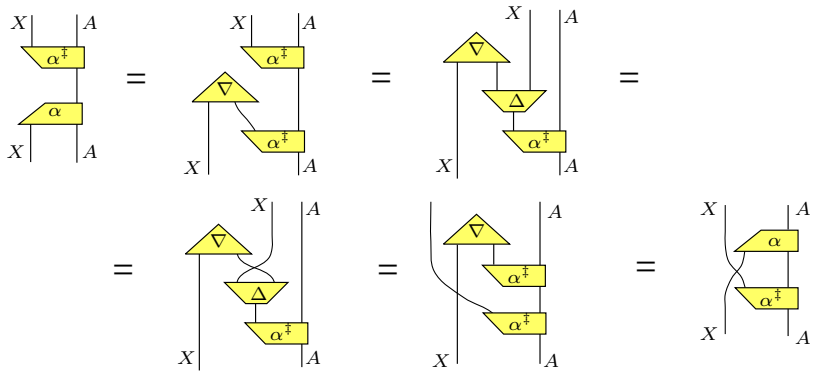
Remarks. The two equations in Thm. 6.4(i) imply each other by applying the dagger. They also imply that

- $X \otimes A \xrightarrow{\alpha} A$ is a retract of $X \otimes X \xrightarrow{\nabla} X$ in the category of X-actions, along the restriction $\alpha^\ddagger : \alpha \rightarrowtail \nabla$, and that
- $A \xrightarrow{\alpha^\ddagger} X \otimes A$ is a retract of $X \xrightarrow{\Delta} X \otimes X$ in the category of X-coactions, along the retraction $\alpha : \Delta \twoheadrightarrow \alpha^\ddagger$.

The Frobenius condition is the special case of both (i) and (ii), since Δ and ∇ are just special actions.

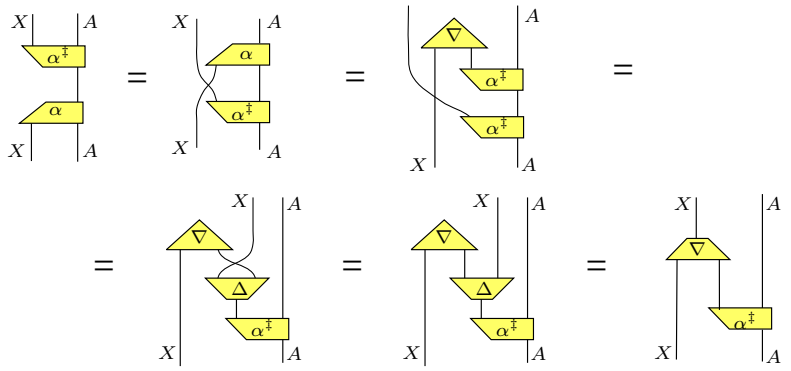
Proof. **(a $\iff$ b)** follows directly from Lemma 6.2. Part (a) of the lemma says that $\alpha(x)$ is idempotent if and only if α is an X-action. Part (b) says that $\alpha(x)$ is self-adjoint if and only if $\alpha = (\varepsilon \otimes A) \circ \alpha^\ddagger$, which is equivalent to $\alpha \circ (x \otimes) = (x^\ddagger \otimes A) \circ \alpha^\ddagger$ by the η-rule, using Thm. 4.3(b).

(a$\Longrightarrow$ii) is proved as follows:



using Lemma 6.2, and the commutativity of Δ.

(ii$\Longrightarrow$i) is a variation on the same theme:



Finally, if the X-action α is normal, then postcomposing (i) with $\top \otimes A$ gives condition 6.2(b), and hence (a).

$\alpha \circ \alpha^\ddagger = \mathrm{id}_A$ is left as an exercise. □

Proposition 6.5 *The category $\mathcal{C}\{x{:}X\}$ of measurements over X is equivalent with the category $\mathcal{C}^{\{X\}}$ of X-actions.*

6.1 Measuring the outcome

In general, the measurement outcome corresponding to a basis vector is the pure projector that represents it. In order to perform the measurement in the first component of $S|x, y\rangle$ from sec. 5, we use a partial representation of this vector.

Lemma 6.6 $\sigma_y(x) = (\nabla_m \otimes \mathrm{id}_n) \circ S|x, y\rangle$ *is a measurement on* $\Xi^{\otimes(m+n)}$ *in* $\mathsf{Rel}_\wp\,[|y\rangle{:}\Xi^{\otimes n}]\{|x\rangle{:}\Xi^{\otimes m}\}$.

Substituting the basis vectors for x in $\sigma_y(x)$ gives the projectors on $\Xi^{\otimes(m+n)}$, from which the information about the period c is extracted like before.

7 Conclusions and future work

Simon's algorithm required three operations:

abstraction: to represent classical functions and classical data in a quantum universe;

transform to a complementary basis: to entangle classical data and make use of quantum parallelism;

measurement: to extract the classical outcomes of quantum computation.

The abstraction operations shape the classical interfaces of quantum computers. Our analysis of the general abstraction operations uncovered a rich structure, that may be of interest beyond quantum computation. Are there other computational resources, besides entanglement, that provide exponential speedup when suitably combined with the general abstraction operations?

The other two operations that we formalized are typically quantum. Complementary bases provide access to entanglement, as the main resource of quantum computation, and thus enable quantum parallelism. The varied interactions among the different classical structures and with measurements give rise to the wealth of quantum algorithms that remain to be explored.

Our abstract model uncovered some abstract entanglement structures, and made them available for quantum computation in non-standard mathematical models. The algorithmic consequences of this semantical result need to be carefully explored.

References

[1] 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.

[2] Samson Abramsky. No-cloning in categorical quantum mechanics. In Simon Gay and Ian Mackie, editors, *Semantical Techniques in Quantum Computation*. Cambridge University Press, 2008. 32 pp, to appear.

[3] Michael Barr and Charles Wells. *Toposes, Triples, and Theories*. Number 278 in Grundlehren der mathematischen Wissenschaften. Springer-Verlag, 1985.

[4] Aurelio Carboni. Matrices, relations, and group representations. *J. of Algebra*, 136:497–529, 1991.

[5] Aurelio Carboni and Robert F.C. Walters. Cartesian bicategories, I. *J. of Pure and Applied Algebra*, 49:11–32, 1987.

[6] Alonzo Church. A formulation of the simple theory of types. *J. of Symbolic Logic*, 5(2):56–68, 1940.

[7] B. Coecke and D. Pavlovic. Quantum measurements without sums. In G. Chen, L. Kauffman, and S. Lamonaco, editors, *Mathematics of Quantum Computing and Technology*. Taylor and Francis, 2007. arxiv.org/quant-ph/0608035.

[8] Bob Coecke and Ross Duncan. Interacting quantum observables. In Luca Aceto, Ivan Damgård, Leslie Ann Goldberg, Magnús M. Halldórsson, Anna Ingólfsdóttir, and Igor Walukiewicz, editors, *ICALP (2)*, volume 5126 of *Lecture Notes in Computer Science*, pages 298–310. Springer, 2008.

[9] Bob Coecke and William Edwards. Toy quantum categories. In Bob Coecke and Prakash Panangaden, editors, *Proceedings of the 2008 QPL-DCM Workshop*, pages 25–35. Springer-Verlag, 2008. arXiv:0808.1037.

[10] Bob Coecke, Dusko Pavlovic, and Jamie Vicary. A new description of orthogonal bases. *Math. Structures in Comp. Sci.*, 2008. 13 pp., to appear, arxiv.org:0810.0812.

[11] D. Dieks. Communication by EPR devices. *Physics Letters A*, 92(6):271–272, 1982.

[12] Bob Coecke Éric Oliver Paquette and Dusko Pavlovic. Classical and quantum structuralism. In Simon Gay and Ian Mackie, editors, *Semantical Techniques in Quantum Computation*. Cambridge University Press, 2008. 42 pp, to appear.

[13] Thomas Fox. Coalgebras and cartesian categories. *Comm. Algebra*, 4(7):665–667, 1976.

[14] Kurt Gödel. Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme. *I. Monatshefte fr Mathematik und Physik*, 38:173–198, 1931.

[15] Sean Hallgren. Polynomial-time quantum algorithms for Pells equation and the principal ideal problem. In *Proceedings of the 34th ACM Symposium on Theory of Computing*, pages 653–658. ACM Press, 2002.

[16] André Joyal and Ross Street. The geometry of tensor calculus I. *Adv. in Math.*, 88:55–113, 1991.

[17] Gregory M. Kelly. *Basic concepts of enriched category theory*. Cambridge University Press, 1982. http://www.tac.mta.ca/tac/reprints/articles/10/tr10.pdf.

[18] Gregory M. Kelly and Miguel L. Laplaza. Coherence for compact closed categories. *J. of Pure and Applied Algebra*, 19:193–213, 1980.

[19] Stephen Cole Kleene. Recursive predicates and quantifiers. *Transactions of the American Mathematical Society*, 53(1):41–73, 1943.

[20] Joachim Kock. *Frobenius Algebras and 2D Topological Quantum Field Theories*, volume 59 of *London Mathematical Society Student Texts*. Cambridge University Press, 2004.

[21] K. Kraus. Complementary observables and uncertainty relations. *Physical Review D*, 35(10):3070–3075, 1987.

[22] Joachim Lambek. From types to sets. *Adv. in Math.*, 36:113–164, 1980.

[23] Joachim Lambek and Philip J. Scott. *Introduction to higher order categorical logic*. Cambridge University Press, New York, NY, USA, 1986.

[24] F. William Lawvere. Adjointness in foundations. *Dialectica*, 23:281–296, 1969.

[25] F.William Lawvere. Ordinal sums and equational doctrines. In *Seminar on Triples, Categories and Categorical Homology Theory*, volume 80 of *Lecture Notes in Mathematics*, pages 141–155. Springer-Verlag, 1969.

[26] Samuel J. Lomonaco and Louis H. Kauffman. Quantum hidden subgroup algorithms: An algorithmic toolkit. In G. Chen, Louis Kauffman, and Samuel Lamonaco, editors, *Mathematics of Quantum Computing and Technology*. Taylor and Francis, 2007.

[27] Saunders Mac Lane. *Categories for the Working Mathematician*. Number 5 in Graduate Texts in Mathematics. Springer-Verlag, 1971.

[28] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, October 2000.

[29] A.K. Pati and S.L. Braunstein. Impossibility of deleting an unknown quantum state. *Nature*, 404:164–165, 2000.

[30] Dusko Pavlovic. Categorical logic of names and abstraction in action calculus. *Math. Structures in Comp. Sci.*, 7:619–637, 1997.

[31] Dusko Pavlovic. Quantum and classical structures in nondeterministic computation. In Peter Bruza, Don Sofge, and Keith van Rijsbergen, editors, *Proceedings of Quantum Interaction 2009*, volume 5494 of *Lecture Notes in Artificial Intelligence*, pages 143–158. Springer Verlag, 2009. arxiv.org:0812.2266.

[32] Dusko Pavlović and Martín Escardó. Calculus in coinductive form. In V. Pratt, editor, *Proceedings. Thirteenth Annual IEEE Symposium on Logic in Computer Science*, pages 408–417. IEEE Computer Society, 1998.

[33] Roger Penrose. Structure of space-time. In C.M. DeWitt and J.A. Wheeler, editors, *Batelle Rencontres, 1967*. Benjamin, 1968.

[34] John Power and Edmund Robinson. Premonoidal categories and notions of computation. *Mathematical. Structures in Comp. Sci.*, 7(5):453–468, 1997.

[35] Peter Selinger. Dagger compact closed categories and completely positive maps. *Electron. Notes Theor. Comput. Sci.*, 170:139–163, 2007.

[36] Peter Selinger. Idempotents in dagger categories: (extended abstract). *Electr. Notes Theor. Comput. Sci.*, 210:107–122, 2008.

[37] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.*, 26(5):1484–1509, 1997.

[38] Daniel R. Simon. On the power of quantum computation. *SIAM J. Comput.*, 26(5):1474–1483, 1997.

[39] D.V. Widder. *An Introduction to Transform Theory*, volume 42 of *Pure and Applied Mathematics*. Academic Press, New York and London, 1971.

[40] William K. Wootters. Quantum measurements and finite geometry, 2004. arXiv.org:quant-ph/0406032.

[41] W.K. Wootters and W.H. Zurek. A single quantum cannot be cloned. *Nature*, 299:802–803, 1982.