What is a Categorical Model of Arrows?

Robert Atkey[1]

LFCS, School of Informatics, University of Edinburgh

Abstract

We investigate what the correct categorical formulation of Hughes' Arrows should be. It has long been folklore that Arrows, a functional programming construct, and Freyd categories, a categorical notion due to Power, Robinson and Thielecke, are somehow equivalent.
In this paper, we show that the situation is more subtle. By considering Arrows wholly within the base category we derive two alternative formulations of Freyd category that are equivalent to Arrows—enriched Freyd categories and indexed Freyd categories. By imposing a further condition, we characterise those indexed Freyd categories that are isomorphic to Freyd categories. The key differentiating point is the number of inputs available to a computation and the structure available on them, where structured input is modelled using comonads.

1 Introduction

Ever since Hughes introduced Arrows [2] and Power, Robinson and Thielecke introduced Freyd categories [11,12] it has been folklore that the two definitions are in some way equivalent. The aim of this paper is to explore the connections between the two definitions. Our main result is that Arrows are in fact more general than Freyd categories. This is due to Arrows allowing two separate inputs to computations, one of which may be structured, while Freyd categories only allow a single input to computations. Generalising further we get *indexed* Freyd categories that allow two inputs, both of which may be structured. Structure on input can be modelled using comonads.

Looking at the definitions of Arrows and Freyd categories it is certainly tempting to think that they are actually the same thing. In Haskell, Arrows are defined as members of the following type class (the actual type class in the Haskell libraries has extra members, but these are definable from the ones we have given):

```
class Arrow a where
    arr   :: (x -> y) -> a x y
    (>>>) :: a x y -> a y z -> a x z
    first :: a x y -> a (x,z) (y,z)
```

[1] Email: bob.atkey@ed.ac.uk

In our categorical definitions we write $\mathrm{Ar}(x, y)$ for `a x y`, *arr* for `arr`, $\ggg$ for `>>>` and *first* for `first`, and we use this notation from now on.

Thus a type operator Ar that takes two arguments is an Arrow if it has definitions for these functions which satisfy some equations (we give these equations in Section 2). A way to read these definitions is to see $\mathrm{Ar}(x, y)$ as the type of "arrows" from x to y. The *arr* function then takes normal functions to arrows and the $\ggg$ function composes two arrows. Thus it seems that each instance of `Arrow` defines a category within Haskell, whose objects are Haskell types and whose arrows are members of $\mathrm{Ar}(x, y)$, and a functor from the category of Haskell types and functions to this new category. The *first* function demands some extra structure on the new category which is used when sequencing arrows.

The definition of Freyd category is beguiling similar. A Freyd category consists of a pair of categories C and K and an identity-on-objects functor $J : C \to K$, where C has finite products, K is symmetric premonoidal and J maps the finite products to the symmetric premonoidal structure. We give the full definition of symmetric premonoidal in Section 3; for now, we just need that it has a functor $- \rtimes z : K(x, y) \to K(x \times z, y \times z)$.

It appears that Freyd categories and Arrows are the same thing: the category C corresponds to the category of Haskell types and functions, the category K is the category described by $\mathrm{Ar}(x, y)$, the functor J is the function *arr*, composition in K is the function $\ggg$ and K's premonoidal structure is the function *first*.

This naive view misses a crucial point: Freyd categories are defined using ordinary categories where there is a *set* of morphisms between objects; Arrows are defined so that there is a Haskell *type* of morphisms between objects. Therefore the correct way to see Arrows is as *enriched* Freyd categories; where the enrichment is over the base category. We recall the definitions of enriched category, functor and so on in Section A.2. We show that Arrows are directly equivalent to enriched Freyd categories in Section 3.2.

However, this view is not particularly enlightening; the assumption that Arrows and Freyd categories are equivalent has persisted for some time, so it seems clear that there must be some direct relationship between the two. Observe that a crucial difference between Freyd categories and Arrows is that a computation described by an Arrow is a morphism in $C(w, \mathrm{Ar}(x, y))$, whereas a computation described by a Freyd category is a morphism in $K(x, y)$. The Arrow computation has additional input w that the Freyd category does not have. To see that this additional input can be significant consider a Freyd category built from a comonad (W, ϵ, δ) so that $K(x, y) = C(Wx, y)$; the analogous Arrow has $C(w, \mathrm{Ar}(x, y)) = C(w, [Wx \to y]) \cong C(w \times Wx, y)$, where W must also be strong. Thus, the arrow computation has access to unstructured input w and structured input x, while the Freyd category computation only has access to structured input x.

To model this situation we generalise from Arrow computations of the form $C(w, \mathrm{Ar}(x, y))$ to a C-indexed category H where $H_w(x, y)$ denotes computations with unstructured input w, possibly structured input x and possibly structured output y. This forms the core of the definition of *indexed Freyd category* that we give in Section 4. Every Arrow gives an indexed Freyd category, but to go back again we must assume that the indexed Freyd category is *closed*, meaning that there

$$\begin{tikzcd}
\begin{array}{c}\text{Freyd}\\\text{Categories}\end{array} \arrow[d, leftrightarrow, "\simeq"] & \begin{array}{c}\text{Closed}\\\text{Freyd}\\\text{Categories}\end{array} \arrow[l, hook'] \arrow[r, leftrightarrow, "\simeq"] \arrow[d, leftrightarrow, "\simeq"] & \text{Strong Monads} \arrow[d, leftrightarrow, "\simeq"] \arrow[dr, leftrightarrow, "\simeq"] & \\
\begin{array}{c}\text{Indexed}\\\text{Freyd}\\\text{Categories}\\\text{+isomorphism}\end{array} \arrow[d, hook, bend right] \arrow[d, phantom, "\dashv"] & \begin{array}{c}\text{Closed}\\\text{Indexed}\\\text{Freyd}\\\text{Categories}\\\text{+isomorphism}\end{array} \arrow[l, hook'] \arrow[r, leftrightarrow, "\simeq"] \arrow[d, hook] & \begin{array}{c}\text{Arrows}\\\text{+isomorphism}\end{array} \arrow[r, leftrightarrow, "="] \arrow[d, hook] & \begin{array}{c}\text{Enriched}\\\text{Freyd}\\\text{Categories}\\\text{+isomorphism}\end{array} \arrow[d, hook] \\
\begin{array}{c}\text{Indexed}\\\text{Freyd}\\\text{Categories}\end{array} \arrow[u, bend right] & \begin{array}{c}\text{Closed}\\\text{Indexed}\\\text{Freyd}\\\text{Categories}\end{array} \arrow[l, hook'] \arrow[r, leftrightarrow, "\simeq"] & \text{Arrows} \arrow[r, leftrightarrow, "="] & \begin{array}{c}\text{Enriched}\\\text{Freyd}\\\text{Categories}\end{array}
\end{tikzcd}$$

Fig. 1. Relationships between notions of computation in this paper

is a representation for morphisms in H:

$$H_w(x, y) \cong C(w, \mathrm{Ar}(x, y)).$$

Indexed Freyd categories that are not closed have more flexibility when it comes to their input. For instance, $H_w(x, y) = C(W(w \times x), y)$ for a comonad W defines an indexed Freyd categories that is not closed.

Not every Arrow demands structure on its input. In particular, Arrows built from strong monads (T, η, μ, τ) have computations as morphisms in $C(w, [x \to Ty]) \cong C(w \times x, Ty)$. In Section 4.2 we examine translations between Freyd categories and indexed Freyd categories and arrive at a characterisation of indexed Freyd categories that are equivalent to Freyd categories, which are those that satisfy the isomorphism:

$$H_{w\times x}(y, z) \cong H_w(x \times y, z).$$

(where the direction $H_w(x \times y, z) \to H_{w\times x}(y, z)$ is always determined). In other words, indexed Freyd categories that do not distinguish between the two inputs. Examples include closed indexed Freyd categories derived from monads and non-closed indexed Freyd categories derived from comonads: $H_w(x, y) = C(W(w\times x), y)$. In Section 5, we translate this requirement back to Arrows, getting the isomorphism:

$$\mathrm{Ar}(x \times y, z) \cong [x \to \mathrm{Ar}(y, z)].$$

(where, again, the direction $[x \to \mathrm{Ar}(y, z)] \to \mathrm{Ar}(x \times y, z)$ is always determined). Using the fact that closed Freyd categories are equivalent to strong monads [12], we can conclude that Arrows that satisfy this condition are equivalent to strong monads.

We summarise the relationships between the various definitions given in this paper in Figure 1. Each node of the diagram is the category of the named objects over a fixed category C and appropriate morphisms. Hooked arrows denote subcategory inclusions, which are all full here. Arrows marked with "$\simeq$" denote equivalences and those marked with "$=$" denote strict equalities. In the bottom left corner, the inclusion functor has a right adjoint which forgets about fibres H_w where $w \neq 1$.

1.1 Related Work

Heunen, Jacobs and Hasuo [1,3] have previously investigated the categorical semantics of Arrows. They take the categorical definition of Arrow to be a monoid in the category of bifunctors $C^{\mathrm{op}} \times C \to V$, where V is symmetric monoidal closed and C is V-enriched. In the case when $V = \mathrm{Set}$, it was proved in [1] that under this definition, Arrows are exactly Freyd categories. This approach has the advantage is being more flexible about what is regarded as "inside" (the category C) the language we are modelling and what is "outside" (the category V). However, matters are complicated by size issues when $V = C$ for cartesian closed C, since, in order to have the colimits necessary to talk about monoids in the category of bifunctors, C can only be a pre-order.

This work was originally inspired by Lindley, Wadler and Yallop's work on the Arrow Calculus [6]. In this calculus there are two typing judgements

$$\Gamma \vdash e : A \qquad\qquad \Gamma; \Delta \vdash M \mathrel{!} A.$$

These correspond to the two categories in the definition of indexed Freyd category; in fact the Arrow calculus is the internal language of closed indexed Freyd categories.

1.2 Overview

In the next section we present the formal definition of Arrows in a cartesian closed category. We also present an alternative but equivalent definition that only requires finite products to be present on the base category. This will be more technically convenient for the rest of the development. In Section 3 we recall the definition of Freyd category and show that Arrows correspond directly to a natural definition of enriched Freyd category. In Section 4 we define indexed Freyd categories and investigate their relationship with normal Freyd categories. Section 5 returns to the connection between Freyd categories and Arrows and characterises those Arrows that are equivalent to closed Freyd categories and hence to strong monads. Section 6 concludes. Auxiliary definitions and notations are given in Appendix A.

2 Arrows

We give two definitions of an Arrow on a category. The first follows the Haskell definition closely, requiring that the base category be cartesian closed. The second definition only requires us to assume that the base category has finite products. We will prove that these two definitions are equivalent, and in Sections 3 and 4 we will use the two alternatives to give two ways of interpreting the statement that "Arrows are Freyd categories".

2.1 Arrows in Cartesian Closed Categories

Let C be a cartesian closed category.

Definition 2.1 An *Arrow* in C consists of a mapping of objects $\mathrm{Ar} : |C| \times |C| \to |C|$ and three families of morphisms:

$$arr_{xy} : [x \to y] \to \mathrm{Ar}(x, y)$$

$$\begin{aligned}\ggg_{xyz} &: \mathrm{Ar}(x,y) \times \mathrm{Ar}(y,z) \to \mathrm{Ar}(x,z)\\ \mathit{first}_{xyz} &: \mathrm{Ar}(x,y) \to \mathrm{Ar}(x \times z, y \times z)\end{aligned}$$

These must satisfy the following nine arrow laws, where we interpret the meanings of these equations using the internal simply-typed λ-calculus of C:

(1) $\mathit{arr}(\mathit{id}) \ggg a = a$
(2) $a \ggg \mathit{arr}(\mathit{id}) = a$
(3) $(a \ggg b) \ggg c = a \ggg (b \ggg c)$
(4) $\mathit{arr}(f;g) = \mathit{arr}(f) \ggg \mathit{arr}(g)$
(5) $\mathit{first}(a) \ggg \mathit{arr}(\pi_1) = \mathit{arr}(\pi_1) \ggg a$
(6) $\mathit{first}(a) \ggg \mathit{arr}(\mathit{id} \times f) = \mathit{arr}(\mathit{id} \times f) \ggg \mathit{first}(a)$
(7) $\mathit{first}(a) \ggg \mathit{arr}(\alpha) = \mathit{arr}(\alpha) \ggg \mathit{first}(\mathit{first}(a))$
(8) $\mathit{first}(\mathit{arr}(f)) = \mathit{arr}(f \times \mathit{id})$
(9) $\mathit{first}(a \ggg b) = \mathit{first}(a) \ggg \mathit{first}(b)$

We have followed a convention to use f and g for variables of function type in C's internal language and a, b, c for variables of arrow type.

We now give four example classes of arrows derived from other structure on C.

- Given a strong monad (T, η, μ, t) on C, we can define its *Kleisli Arrow* by $\mathrm{Ar}(x,y) = [x \to Ty]$. The three families of morphisms are defined as follows:

  $$\begin{aligned}\mathit{arr}_{xy}(f) &= f;\epsilon\\ a \ggg_{xyz} b &= a; Tb; \mu\\ \mathit{first}_{xyz}(a) &= a \times z; \sigma_{T(y)z}; \tau; T\sigma_{zy}\end{aligned}$$

  where $\sigma_{xy} : x \times y \to y \times x$ is the symmetry operation. This is the traditional definition for computational effects [9].

- Given a strong comonad (W, ϵ, δ, t) on C, we can define its *CoKleisli Arrow* by $\mathrm{Ar}(x,y) = [Wx \to y]$. In this case, the three families of morphisms are defined as follows:

  $$\begin{aligned}\mathit{arr}_{xy}(f) &= \epsilon; f\\ a \ggg_{xyz} b &= \delta; Wa; b\\ \mathit{first}_{xyz}(a) &= \langle W\pi_1, \epsilon; \pi_2\rangle; a \times z\end{aligned}$$

  The strength is used to define the "internal" action of $W : [x \to y] \to [Wx \to Wy]$. Comonads have been used to give semantics to dataflow computations [13].

- Given a strong comonad $(W, \epsilon, \delta, t_W)$ and a strong monad (T, η, μ, t_T) that have a distributive law $\lambda : WT \to TW$, then we can define their *BiKleisli Arrow* $\mathrm{Ar}(x,y) = [Wx \to Ty]$. Uustalu and Vene use this construction to give a semantics to partial dataflow programs [13].

- Given an applicative functor $(I, \mathit{pure}, \circledast)$ [8] its *Static Arrow* is given by $\mathrm{Ar}(x,y) = I([x \to y])$. The three families of morphisms are defined as:

  $$\begin{aligned}\mathit{arr}_{xy}(f) &= \mathit{pure}(f)\\ a \ggg_{xyz} b &= \mathit{pure}(;') \circledast a \circledast b\\ \mathit{first}_{xyz}(a) &= \mathit{pure}(\times \mathit{id}_z) \circledast a\end{aligned}$$

where $(;')$ is the curried version of the composition operator.

Lemma 2.2 *Given an Arrow, we can define another transformation* $\mathit{second}_{xyz} : \mathrm{Ar}(x,y) \to \mathrm{Ar}(z \times x, z \times y)$*, that satisfies the following laws:*

$$\mathit{second}(a) = \mathit{arr}(\sigma) \ggg \mathit{first}(a) \ggg \mathit{arr}(\sigma) \tag{10}$$

$$\mathit{second}(a) \ggg \mathit{arr}(\pi_2) = \mathit{arr}(\pi_2) \ggg a \tag{11}$$

$$\mathit{second}(a) \ggg \mathit{arr}(f \times \mathit{id}) = \mathit{arr}(f \times \mathit{id}) \ggg \mathit{second}(a) \tag{12}$$

$$\mathit{second}(\mathit{second}(a)) \ggg \mathit{arr}(\alpha) = \mathit{arr}(\alpha) \ggg \mathit{second}(a) \tag{13}$$

$$\mathit{second}(\mathit{arr}(f)) = \mathit{arr}(\mathit{id} \times f) \tag{14}$$

$$\mathit{second}(a \ggg b) = \mathit{second}(a) \ggg \mathit{second}(b) \tag{15}$$

Proof. Define $\mathit{second}_{wxyz}(a) = \mathit{arr}(\sigma_{zx}) \ggg \mathit{first}(a) \ggg \mathit{arr}(\sigma_{yz})$, where $\sigma_{xy} : x \times y \to y \times x$ is the symmetry function; this automatically satisfies equation (10). Equation (11) follows from arrow laws (4) and (5) and the fact that $\sigma; \pi_1 = \pi_2$ and vice versa. Likewise, equations (12), (13), (14) and (15) follow from the corresponding law for *first* and the properties of the symmetric monoidal structure defined in terms of C's internal language. □

2.2 Arrows in Categories with Finite Products

We now give an alternative definition of Arrow, but this time only assuming that the base category C has (chosen) finite products. We will show that, when C is cartesian closed, this definition is equivalent to the previous one.

In order to state this definition, we make use of the indexed category $\widehat{C} : C^{\mathrm{op}} \to \mathrm{Cat}$, defined in Section A.3.

Definition 2.3 An Arrow on a category C with finite products is given by a mapping of objects $\mathrm{Ar} : |C| \times |C| \to |C|$ and three transformations of homsets, all natural in w:

$$\begin{aligned}
\mathit{arr}_{wxy} &: \widehat{C}_w(x,y) \to C(w, \mathrm{Ar}(x,y)) \\
\ggg_{wxyz} &: C(w, \mathrm{Ar}(x,y)) \times C(w, \mathrm{Ar}(y,z)) \to C(w, \mathrm{Ar}(x,z)) \\
\mathit{first}_{wxyz} &: C(w, \mathrm{Ar}(x,y)) \to C(w, \mathrm{Ar}(x \times z, y \times z))
\end{aligned}$$

These transformations must obey the same laws (1)-(9) as for arrows, where the equations are interpreted as equalities between C morphisms generated by the above transformations and the structure of $\widehat{C}$. In those equations we use f, g for morphisms in $\widehat{C}_w(x,y)$ and a, b, c for morphisms in $C(w, \mathrm{Ar}(x,y)$.

Theorem 2.4 *For a cartesian closed category, to give an Arrow as in Definition 2.1 is exactly to give one as in Definition 2.3.*

Proof. Taking the three transformations in Definition 2.3 and using the isomorphisms of homsets due to the cartesian closure, we see that they are in bijective correspondence with triples of transformations of the form

$$\begin{aligned}
\mathit{arr}'_{wxy} &: C(w, [x \to y]) \to C(w, \mathrm{Ar}(x,y)) \\
\ggg'_{wxyz} &: C(w, \mathrm{Ar}(x,y) \times \mathrm{Ar}(y,z)) \to C(w, \mathrm{Ar}(x,z)) \\
\mathit{first}'_{wxyz} &: C(w, \mathrm{Ar}(x,y)) \to C(w, \mathrm{Ar}(x \times z, y \times z))
\end{aligned}$$

all natural in w. By Yoneda, these are in bijective correspondence with the three families of arrows required by Definition 2.1. It remains to check that the required equations hold in both directions. This involves straightforward diagram chasing for each equation. For example, for the associativity of Arrow composition, equation (3), the diagram is:

$$\begin{tikzcd}
C\left(v, \begin{array}{c}\mathrm{Ar}(w,x)\times\\ \mathrm{Ar}(x,y)\times\\ \mathrm{Ar}(y,z)\end{array}\right) \arrow[rrr, "-;\ggg\times id"] \arrow[dr, "\Pi^{-1};\Pi^{-1}\times id=\Pi^{-1};id\times\Pi^{-1}"] \arrow[ddd, "-;id\times\ggg"'] & & & C\left(v, \begin{array}{c}\mathrm{Ar}(w,y)\times\\ \mathrm{Ar}(y,z)\end{array}\right) \arrow[dl, "\Pi^{-1}"] \arrow[ddd, "-;\ggg"] \\
& \begin{array}{c}C(v,\mathrm{Ar}(w,x))\times\\ C(v,\mathrm{Ar}(x,y))\times\\ C(v,\mathrm{Ar}(y,z))\end{array} \arrow[r, "\ggg'\times id"] \arrow[d, "id\times\ggg'"] & \begin{array}{c}C(v,\mathrm{Ar}(w,y))\times\\ C(v,\mathrm{Ar}(y,z))\end{array} \arrow[ddr, "\ggg'"] & \\
& \begin{array}{c}C(v,\mathrm{Ar}(w,x))\times\\ C(v,\mathrm{Ar}(x,z))\end{array} \arrow[drr, "\ggg'"] & & \\
C\left(v, \begin{array}{c}\mathrm{Ar}(x,y)\times\\ \mathrm{Ar}(y,z')\end{array}\right) \arrow[ur, "\Pi^{-1}"] \arrow[rrr, "-;\ggg"] & & & C(v,\mathrm{Ar}(w,z))
\end{tikzcd}$$

where $\Pi : C(w,x)\times C(w,y)\cong C(w,x\times y)$. The outer diagram is the associativity law for Definition 2.1 and the inner diagram is associativity for Definition 2.3. The four diagrams around the edge all commute by the naturality of Π^{-1} and the correspondence between $\ggg$ and $\ggg'$. Hence the inner diagram commutes if the outer ones does, and vice versa. □

The following lemma states that Definition 2.3 can also have an additional *second* transformation defined upon it. The proof is straightforward: one can either follow the proof of the previous theorem and translate the *second* transformation defined in the proof of Lemma 2.2, or one can directly follow the proof of Lemma 2.2 and define *second* directly from the *first*.

Lemma 2.5 *Given an Arrow, we can define another transformation* $second_{xyz} : C(w,\mathrm{Ar}(x,y)) \to C(w,\mathrm{Ar}(z\times x, z\times y))$*, that satisfies the laws (10) to (15) given above.*

3 Arrows and Freyd Categories

In this section we examine the relationship between Arrows and Freyd categories. We first recall the definition of a (closed) Freyd category and the relationship with strong monads. We then investigate how the definition of Arrow in a CCC (Definition 2.1) can be seen to be a Freyd category, where we arrive at a natural definition of an *enriched* Freyd category. In the next section we will show that enriched Freyd categories and Freyd categories are not the same thing.

3.1 Freyd Categories

We first recall the definition of ordinary Freyd categories, following [11,12].

Definition 3.1 Given a category K, a *binoidal structure* on K consists of two families of functors $x \ltimes - : K \to K$ and $- \rtimes x : K \to K$, where $x \in |K|$, such that $x \ltimes y = x \rtimes y$, which we write as $x \otimes y$.

A morphism $f : x \to x'$ of a binoidal category K is *central* if, for all $g : y \to y'$, $f \rtimes y; x' \ltimes g = x \ltimes g; f \rtimes y'$ and symmetrically.

Definition 3.2 Given a binoidal category K, *symmetric premonoidal structure* on K consists of an object I and four natural transformations $\alpha : (x \otimes y) \otimes z \to x \otimes (y \otimes z)$, $\lambda : x \otimes I \to x$, $\rho : I \otimes x \to x$ and $\sigma : x \otimes y \to y \otimes x$, whose components are all central and who satisfy the usual coherence conditions for symmetric monoidal structure [7].

Note that any category with a choice of finite products and a terminal object (any indeed any symmetric monoidal category) is symmetric premonoidal.

Definition 3.3 A *Freyd category* consists of a category C with finite products, a symmetric premonoidal category K and an identity-on-objects functor that exactly preserves symmetric premonoidal structure and such that $J(f)$ is always central.

Definition 3.4 A *closed* Freyd category is a Freyd category with a right adjoint to $J - \times y$ for every y:

$$\Lambda : K(x \times y, z) \cong C(x, [y \rightsquigarrow z])$$

We use the notation $[x \rightsquigarrow y]$ to distinguish from the cartesian closed internal hom $[x \to y]$.

The equivalences we consider in the next section will only be up to isomorphism of Freyd categories, so we must define morphisms of Freyd categories.

Definition 3.5 Given two Freyd categories $J^1 : C \to K^1$ and $J^2 : C \to K^2$, a morphism between them is a functor $F : K^1 \to K^2$ that strictly preserves symmetric premonoidal structure and satisfies $F(J^1(f)) = J^2(f)$. Freyd categories and their morphisms form a category $\mathrm{Freyd}(C)$. Closed Freyd categories form a category $\mathrm{CFreyd}(C)$.

The following proposition [5] establishes a link between closed Freyd categories and strong monads.

Proposition 3.6 *For a cartesian closed category C, the category* $\mathrm{CFreyd}(C)$ *is equivalent to the category of strong monads and strong monad morphisms.*

3.2 Enriched Freyd Categories

We start with a cartesian closed category C with an Arrow $(\mathrm{Ar}, \mathit{arr}, \ggg, \mathit{first})$ as in Definition 2.1, and work towards a suitable definition of Freyd category that matches. Intuitively, the *first* transformations, and the derived *second* transformations, should correspond to the required symmetric premonoidal structure, but on what category? And what corresponds to the functor J?

The answer can be seen by looking at the first two parts of Definition 2.1 and comparing them to the definition of enriched category (see Appendix A.2 or Kelly [4]). Given a category C with finite products, a C-enriched category D consists of a collection of objects $|D|$, a mapping $D : |D| \times |D| \to |C|$ (the hom-objects) and a pair of transformations $id^D_x : 1 \to D(x,x)$ and $comp^D_{xyz} : D(x,y) \times D(y,z) \to D(x,z)$ which satisfy the evident equations for identities and associativity. It is standard that if C is cartesian closed, then by taking the hom-objects to be $[x \to y]$ we can see C as being C-enriched.

Looking at Definition 2.1, there is another obvious choice for hom-objects with C's objects: take the hom-objects to be $\mathrm{Ar}(x,y)$ and use $arr(id)$ for identities and $\ggg$ for composition. It is easy to see that Arrow laws (1), (2), (3) imply the laws for a C-enriched category and vice versa. Thus we have two C-categories with the same objects as C: $\underline{C}$, enriched by $[x \to y]$; and $\underline{K}$, enriched by $\mathrm{Ar}(x,y)$.

A C-enriched functor is given by a mapping of objects $F : |D| \to |D'|$ and a transformation of hom-objects $D(x,y) \to D'(Fx,Fy)$ that preserves identities and composition. We can use the transformation arr to define a C-functor from $\underline{C}$ to $\underline{K}$, which will do for our enriched analogue of J. The Arrow law (4) and our definition of identities as $arr(id)$ implies the laws for a C-functor and vice versa.

So we now have that the $(\mathrm{Ar}, arr, \ggg)$ part of the definition of an Arrow on a category C is equivalent to having a pair of C-categories with the same objects as C and an identity-on-objects functor between them. It remains to devise a suitable analogue of symmetric premonoidal structure for C-categories, and to show that it corresponds exactly to the transformation *first*.

The definition of binoidal structure carries over directly from the non-enriched case simply by replacing "functor" by "C-functor" everywhere. Binoidal structure on $\underline{C}$ is given directly by C's finite products, and binoidal structure on $\underline{K}$ is given by $f \rtimes x = \mathit{first}(f)$ and $x \ltimes f = \mathit{second}(f)$. By Arrow laws (8), (9), (14) and (15) these give C-functors.

A slightly tricky point is the definition of centrality for C-enriched binoidal structure because we cannot say "for all morphisms...". We deal with this by requiring that all arrows produced by arr are central, in the following sense:

$$\begin{tikzcd}
{[x \to x'] \times \mathrm{Ar}(y,y')} \arrow[r, "\sigma"] \arrow[d, "arr\times id"'] & {\mathrm{Ar}(y,y') \times [x \to x']} \arrow[d, "id\times arr"] \\
{\mathrm{Ar}(x,x') \times \mathrm{Ar}(y,y')} \arrow[d, "(-\rtimes y)\times(x'\ltimes -)"'] & {\mathrm{Ar}(y,y') \times \mathrm{Ar}(x,x')} \arrow[d, "(x\ltimes -)\times(-\rtimes y')"] \\
{\mathrm{Ar}(x\times y, x'\times y) \times \mathrm{Ar}(x'\times y, x'\times y')} \arrow[dr, "\ggg"'] & {\mathrm{Ar}(x\times y, x\times y') \times \mathrm{Ar}(x\times y', x'\times y')} \arrow[d, "\ggg"] \\
 & {\mathrm{Ar}(x\times y, x'\times y')}
\end{tikzcd}$$

and symmetrically for the $[x \to y]$ component "on the bottom".

It is easy to see that by Arrow laws (6), (8), (12) and (14) that Arrows satisfy these diagrams, and conversely. The final parts we need, that the symmetric monoidal structure isomorphisms are natural is covered by Arrow laws (5), (7) and (10).

In summary, we have proved:

Theorem 3.7 *For a cartesian closed category C, to give an Arrow as in Definition*

2.1 is to give a C-enriched Freyd category $J : \underline{C} \to \underline{K}$.

An obvious step now would be to go on and define a notion of closure for C-enriched Freyd categories. We defer this step until Section 5.

4 Indexed Freyd Categories

We now start from an Arrow in a category with finite products (Definition 2.3) and examine how it can been seen to be analogous to the structure of a Freyd category. The structure that we finish up with, which we call *indexed Freyd categories*, will be more general than that of Arrows. We then investigate the relationship between indexed Freyd categories and Freyd categories and give an extra condition on indexed Freyd categories that makes them isomorphic to Freyd categories.

4.1 Indexed Freyd Categories

As in the previous section we seek a category on which we can find a premonoidal structure to use as our analogue of the codomain category of a Freyd category. The crucial step is to take the natural transformation $arr : \widehat{C}_w(x, y) \to C(w, \mathrm{Ar}(x, y))$ from Definition 2.3 and see it as a functor between C-indexed categories $\widehat{C}$ and H, where $|H_w| = |C|$ and $H_w(x, y) = C(w, \mathrm{Ar}(x, y))$. H is a C indexed category with identities in the fibres given by $arr(id)$ and composition by $\ggg$. By the Arrow laws (1), (2) and (3) the fibres are well-defined categories. Re-indexing is accomplished by precomposition in C. The mapping $arr : \widehat{C}_w(x, y) \to H_w(x, y)$ defines an identity-on-objects indexed functor $J : \widehat{C} \to H$ by the naturality of arr and Arrow law (4).

Each of the fibres H_w can be given premonoidal structure: we define $a \rtimes x = \mathit{first}(a)$ and $x \ltimes a = \mathit{second}(a)$; by arrow laws (8), (9), (14) and (15), these are well-defined functors. By laws (8), (6), (14) and (12), every morphism of the form $arr(f)$ is central; hence, we can define the premonoidal structure transformations via arr by those in $\widehat{C}_w$. Laws (5), (7), (10), (11) and (13) ensure that these are natural. By the naturality of *first* and *second*, the re-indexing functors H_f preserve the premonoidal structure.

Each fibre of $\widehat{C}$ automatically has premonoidal structure derived from the chosen finite products and the indexed functor J defined above preserves the premonoidal structure by laws (8) and (14) and the sufficient condition for centrality in the fibres H_w given above.

We consolidate the structure we have induced into a definition:

Definition 4.1 Given a category C with finite products, an *indexed Freyd category* consists of an indexed category $H : C^{\mathrm{op}} \to \mathrm{Cat}$ and an identity-on-objects indexed functor $J : \widehat{C} \to H$, such that:

(i) Each H_w has symmetric premonoidal structure;

(ii) All re-indexing functors $H_f : H_{w'} \to H_w$ are identity-on-objects and strict symmetric premonoidal;

(iii) The functors $J_w : \widehat{C}_w \to H_w$ are strict symmetric premonoidal and every $J_w(f)$ is central.

We have shown that it is possible to go from an Arrow on a category with finite products to an indexed Freyd category, but it is not yet possible to go back the other way: there is no way to obtain an object of C that represents the arrow! We fix this by providing a further definition:

Definition 4.2 A *closed* indexed Freyd category is an indexed Freyd category $J : \widehat{C} \to H$ equipped with, for every pair of objects x and y, an object $[x \rightsquigarrow y]$ and an isomorphism

$$\Lambda_{wxy} : H_w(x, y) \cong C(w, [x \rightsquigarrow y])$$

natural in w.

An example of a non-closed indexed Freyd category is given by a category C with finite products and a functor $F : C \to C$ with natural transformation $\alpha_x : Fx \to x$ and a strong comonad (W, ϵ, δ, t). Define $H_w(x, y) = C(Fw \times Wx, y)$. It is easy to see that this gives an indexed Freyd category that is not, in general, closed.

Proposition 4.3 *Given a category C with finite products, every Arrow gives rise to a closed indexed Freyd category and vice versa.*

Proof. In the forward direction it remains to show that indexed Freyd categories arising from Arrows are closed: define the objects $[x \rightsquigarrow y] = \mathrm{Ar}(x, y)$; the isomorphism is trivial since $H_w(x, y) = C(w, \mathrm{Ar}(x, y))$.

Conversely, we can take any indexed Freyd category and obtain an arrow on the base category C. Define $\mathrm{Ar}(x, y) = [x \rightsquigarrow y]$ and:

$$\begin{aligned} \mathit{arr}(f) &= \Lambda(J(f)) \\ a \ggg b &= \Lambda(\Lambda^{-1}(a); \Lambda^{-1}(b)) \\ \mathit{first}(a) &= \Lambda(\Lambda^{-1}(a) \rtimes x) \end{aligned}$$

The laws are now easy to check. □

The previous proposition states that we have two well-defined translations: one from Arrows to closed indexed Freyd categories, and one in the opposite direction. We now show that these two translations are mutually inverse, up to isomorphism. To state exactly what we mean by isomorphisms here, we define morphisms between Arrows on a base category and morphisms between indexed Freyd categories on a base category.

Definition 4.4 Let $(\mathrm{Ar}^1, \mathit{arr}^1, \ggg^1, \mathit{first}^1)$ and $(\mathrm{Ar}^2, \mathit{arr}^2, \ggg^2, \mathit{first}^2)$ be two Arrows on C in the sense of Definition 2.3. A morphism between them consists of a family of morphisms $F_{xy} : \mathrm{Ar}^1(x, y) \to \mathrm{Ar}^2(x, y)$ such that:

$$\begin{aligned} \mathit{arr}^1(f); F &= \mathit{arr}^2(f) \\ (a \ggg^1 b); F &= (a; F) \ggg^2 (b; F) \\ \mathit{first}^1(a); F &= \mathit{first}^2(a; F) \end{aligned}$$

Arrows and Arrow morphisms form a category $\mathrm{Arrows}(C)$.

Definition 4.5 Let $J^1 : \widehat{C} \to H^1$ and $J^2 : \widehat{C} \to H^2$ be two indexed Freyd categories. A morphism between them is an indexed functor $F : H^1 \to H^2$ that strictly preserves symmetric premonoidal structure and satisfies $F_w(J^1_w(f)) = J^2_w(f)$.

Indexed Freyd categories and morphisms form a category IdxFreyd(C). Closed indexed Freyd categories and morphisms form a category CIdxFreyd(C).

Theorem 4.6 *The categories* Arrow(C) *and* CIdxFreyd(C) *are equivalent.*

Proof. We must first check that the two translations defined above can be extended to be functors. Given an Arrow morphism $F : \mathrm{Ar}^1(x, y) \to \mathrm{Ar}^2(x, y)$ we need an indexed Freyd category morphism that will map $C(w, \mathrm{Ar}^1(x, y))$ to $C(w, \mathrm{Ar}^2(x, y))$, which is easily given by composition. In the opposite direction, given an closed indexed Freyd category morphism $F : H_1 \to H_2$, define an Arrow morphism via Yoneda on the composite:

$$C(w, [x \rightsquigarrow^1 y]) \cong H^1_w(x, y) \xrightarrow{F_w} H^2_w(x, y) \cong C(w, [x \rightsquigarrow^2 y]).$$

We now check that this is an equivalence. Starting from an Arrow, translating to an indexed Freyd category and then back again, it is easy to check that we end up exactly where we started. Starting from a closed indexed Freyd category (J, H) translating to an Arrow and back again we get a closed indexed Freyd category where $H'_w(x, y) = C(w, \mathrm{Ar}(x, y))$ and $J'(f) = \Lambda(Jf)$. This is isomorphic to (J, H) by Λ. □

4.2 Indexed Freyd Categories and Freyd Categories

We now examine the relationship between indexed Freyd categories and Freyd categories. It turns out that Freyd categories are embedded co-reflectively in indexed Freyd categories. First, we set up the two translations.

Lemma 4.7 *There is a functor* $M : \mathrm{IdxFreyd}(C) \to \mathrm{Freyd}(C)$.

Proof. Given an indexed Freyd category $J : \widehat{C} \to H$ we obtain an ordinary Freyd category $J_K : C \to K$ by setting $K(x, y) = H_1(x, y)$ and taking J_K to be the composite of the isomorphism between C and $\widehat{C}_1$ and the functor J_1. This definition makes K symmetric premonoidal because H_1 is, and J_K preserves symmetric premonoidal structure because the two components it is built from do. Indexed Freyd category morphisms $F : H^1 \to H^2$ give Freyd category morphisms by restriction to $F_1 : K^1 = H^1_1 \to H^2_1 = K^2$. □

Lemma 4.8 *There is a functor* $N : \mathrm{Freyd}(C) \to \mathrm{IdxFreyd}(C)$.

Proof. Assume a Freyd category $J : C \to K$. Following the construction of $\widehat{C}$ from C given in Section A.3, we construct $\widehat{K}$ from K by considering the comonads $W_w = w \ltimes -$ induced by objects w. The counit and comultiplication are given by $J(\pi_2) : W_w x \to x$ and $J(\langle \pi_1, id) : W_w x \to W_w W_w x$ respectively; these are exactly the counit and comultiplication of the comonad we defined in C, via J. Again, every morphism $f : w \to w'$ in C gives a comonad morphism $W_f : W_w \to W_{w'}$. We use these to construct a C-indexed category $\widehat{K}$ where each fibre $\widehat{K}_w$ has the same objects as C and homsets $\widehat{K}_w(x, y) = K(W_w x, y) = K(w \times x, y)$. Similar to the definition of a choice of finite product structure on the fibres of $\widehat{C}$, symmetric premonoidal structure is definable on each of the fibres of $\widehat{K}$: given $f \in \widehat{K}_w(x, y)$ define $f \mathbin{\widehat{\rtimes}} z = \alpha^{-1}; f \rtimes z$. The functor $J : C \to K$ straightforwardly induces an indexed functor $\widehat{J} : \widehat{C} \to \widehat{K}$ which is easily checked to preserve the premonoidal structure. Given a Freyd category morphism $F : K^1 \to K^2$ we get an indexed Freyd category morphism defined by $H^1_w(x,y) = K^1(w \times x, y) \xrightarrow{F} K^2(w \times x, y) = H^2_w(x,y)$. □

Lemma 4.9 *The functor N is an embedding.*

Proof. We must check that N is injective on objects and is faithful. Injectivity on objects is obvious: if $\widehat{K^1} = \widehat{K^2}$, then $K^1 = K^2$. Faithfulness also follows. □

One can immediately see that there is not much hope of getting an equivalence here: the translation from indexed Freyd category to Freyd category only makes use of the fibre H_1, not any of the others. Nevertheless, we play through what happens by looking at the back and forth translations.

Starting from a Freyd category, applying N and then M to get back to a Freyd category we end up the following definitions: $K'(x,y) = K(1 \times x, y)$ and $J'(f) = J(\pi_2; f)$. It is clear that there are Freyd category morphisms making this isomorphic to the original Freyd category.

If we start from an indexed Freyd category, things are different. Given an indexed Freyd category $J : \widehat{C} \to H$, going via Freyd categories gives us $H'_w(x,y) = H_1(w \times x, y)$ and $J_w(f) = J_1(\pi_2; f)$. It is not the case in general that H' is going to be isomorphic to H: consider the indexed Freyd category constructed from a comonad (W, ϵ, δ) which has $H_w(x,y) = C(w \times Wx, y)$. Then $H'_w(x,y) = C(1 \times W(w \times x), y)$. These are clearly not isomorphic.

However, it is the case that there is a morphism from $H' \to H$ given by mapping $a \in H_1(w \times x, y)$ to the following arrow in H_w:

$$x \xrightarrow{J(\widehat{id})} w \times x \xrightarrow{H_!(a)} y$$

where $! : w \to 1$ is the unique morphism. Some straightforward checking ensures that this is a natural transformation $\varepsilon : NM \to Id$ in IdxFreyd(C).

Theorem 4.10 *The category* Freyd(C) *co-reflectively embeds into* IdxFreyd(C).

Proof. It remains to show that M is right adjoint to N. The unit η of the adjunction is the isomorphism between MN and Id given above and the counit is the natural transformation $\varepsilon : NM \to Id$ given above. Some lengthy but tedious checking ensures that these obey the triangular identities for adjunctions. □

Given the above counterexample to the counit of the adjunction being an isomorphism, we can see the essential difference between Freyd categories and indexed Freyd categories. In a Freyd category, computations, represented by members of $K(x,y)$ have one input, which may or may not be structured: in the case of a Freyd category built from a comonad ($K(x,y) = C(Wx,y)$), it is structured; in the case of a Freyd category built from a monad ($K(x,y) = C(x,Ty)$) it is unstructured. Indexed Freyd categories on the other hand have two inputs: computations in $H_w(x,y)$ have access to an input of type w and an input of type x, either of which may be structured. In the case of closed indexed Freyd categories, the w input must be unstructured.

In the case when the indexed Freyd category is built from a monad, so that $H_w(x,y) = C(w \times x, Ty)$, there is no distinction between the two kinds of input.

We then have an inverse to the counit ε, giving an isomorphism:

$$\varepsilon : H_1(w \times x, y) \cong H_w(x, y)$$

An equivalent, and nicer, formulation is to state this with an object in place of 1:

$$H_w(x \times y, z) \cong H_{w\times x}(y, z) \tag{16}$$

where the $H_w(x \times y, z) \to H_{w\times z}(y, z)$ direction is given by $a \mapsto J(\widehat{\pi_2 \times y}); H_{\pi_1}(a)$. We will show in the next section that closed indexed Freyd categories that satisfy this isomorphism are equivalent to strong monads.

Theorem 4.11 *The subcategory of* IdxFreyd(C) *whose objects satisfy (16) is equivalent to* Freyd(C)*.*

Proof. We have exactly demanded the additional property required for the functors M and N to be an equivalence. □

We can extend this to an equivalence between closed indexed Freyd categories that satisfy (16) and closed Freyd categories:

Theorem 4.12 *The subcategory of* CIdxFreyd(C) *whose objects satisfy (16) is equivalent to* CFreyd(C)*.*

Note that the translation M : IdxFreyd$(C) \to$ Freyd(C) does not extend to the categories with closure. For example, if we start from a closed indexed Freyd category derived from a CoKleisli arrow, then $H_w(x, y) = C(w, [Wx \to y])$ and the derived Freyd category via M has $K(x, y) = H_1(x, y) = C(1, [Wx \to y]) \cong C(Wx, y)$. It is clear that this Freyd category is not in general closed. A higher-level view of this failure is that the required isomorphism for closed Freyd categories:

$$K(x \times y, z) \cong C(x, [y \rightsquigarrow z])$$

splits the input of the computation in an arbitrary place, meaning that any structure the input has must be very weak.

5 Arrows and Freyd Categories, Revisited

We know from Proposition 3.6 that there is an equivalence between closed Freyd categories and strong monads. Given the equivalence between closed Freyd categories and closed indexed Freyd categories satisfying isomorphism (16), it is natural to ask what this means in terms of arrows.

Given an Arrow on a cartesian closed category C, such that the indexed Freyd category derived from it satisfies isomorphism (16), we can derive the following isomorphism:

$$\begin{aligned} C(w, \mathrm{Ar}(x \times y, z)) &\cong H_w(x \times y, z) \\ &\cong H_{w\times x}(y, z) \\ &\cong C(w \times x, \mathrm{Ar}(y, z)) \\ &\cong C(w, [x \to \mathrm{Ar}(y, z)]) \end{aligned}$$

natural in w. Thus, via Yoneda, we have an isomorphism

$$\mathrm{Ar}(x \times y, z) \cong [x \to \mathrm{Ar}(y, z)] \tag{17}$$

where the $\mathrm{Ar}(x \times y, z) \to [x \to \mathrm{Ar}(y,z)]$ direction is $a \mapsto \lambda x.arr(\lambda y.(x,y)) \ggg a$ in C's internal language.

Theorem 5.1 *The subcategory of* Arrow(C) *given by Arrows that support isomorphism (17) is equivalent to* CFreyd(C)*.*

Proof. Observe that Arrows that support isomorphism (17) are equivalent to closed indexed Freyd categories that support isomorphism (16) by a restriction of Theorem 4.6. Hence by Theorem 4.12 the theorem follows. □

Corollary 5.2 *The subcategory of* Arrow(C) *given by Arrows that support isomorphism (17) is equivalent to the category of strong monads on C and strong monad morphisms.*

We can now define closed C-enriched Freyd categories as those that support isomorphism (17).

Corollary 5.3 *Closed C-enriched Freyd categories are equivalent to strong monads.*

We can recover the monad associated with an Arrow that supports isomorphism (17) by setting $Tx = \mathrm{Ar}(1,x)$. The isomorphism is essential in defining the multiplication of the monad. Arrows that are really monads have been identified in Haskell [2] as those that are members of the `ArrowApply` class:

```
class Arrow a => ArrowApply a where
   app :: a (a b c, b) c
```

That is, there is an Arrow $\mathrm{Ar}(\mathrm{Ar}(x,y) \times x, y)$, performing internal evaluation of Arrows. This is derivable from isomorphism (17) by taking the identity $[\mathrm{Ar}(y,z) \to \mathrm{Ar}(y,z)]$ backwards across the isomorphism (this is the direction that is not always definable).

6 Conclusions

We have shown that there is more to the connection between Freyd categories and Arrows than initially meets the eye. The crucial observation is that Arrows are defined completely within a base category, and so have more flexibility than Freyd categories. Investigating the connection between Freyd categories and Arrows has led to the interesting definition of indexed Freyd category, and the notion of "first-order" arrows—indexed Freyd categories that are not closed.

In future work we wish to establish the connection between Arrows/indexed Freyd categories and other notions of computation such as comonads and applicative functors in the same way as we have for strong monads in this work. We also want to investigate additional structure on Arrows such as recursion (the `ArrowLoop` type class) and coproducts (the `ArrowChoice` type class) and see what happens to the related indexed Freyd categories. Also, it would be interesting to consider freely adding closure to indexed Freyd categories, following Power's free addition of closure to Freyd categories [10]. The additional theory developed by Jacobs and Hasuo in [3], which relates the definition of Arrows as monoids in categories of bifunctors to Eilenberg-Moore and Kleisli constructions also suggests further work.

Acknowledgements

I wish to thank Sam Lindley, Philip Wadler and Jeremy Yallop for useful discussions and insights into Arrows and related notions. This work was funded by the ReQueST grant (EP/C537068) from the Engineering and Physical Sciences Research Council.

References

[1] Heunen, C. and B. Jacobs, *Arrows, like Monads, are Monoids*, Electr. Notes Theor. Comput. Sci. **158** (2006), pp. 219–236.

[2] Hughes, J., *Generalising monads to arrows*, Science of Computer Programming **37** (2000), pp. 67–111. URL http://www.cs.chalmers.se/~rjmh/Papers/arrows.ps

[3] Jacobs, B. and I. Hasuo, *Freyd is Kleisli, for Arrows*, in: C. McBride and T. Uustalu, editors, *Proceedings of Workshop on Mathematically Structured Functional Programming*, BCS eWiC, 2006.

[4] Kelly, G. M., "Basic Concepts of Enriched Category Theory," Number 64 in London Mathematical Society Lecture Notes, Cambridge University Press, 1982, available as Reprints in Theory and Applications of Categories, No. 10, 2005.

[5] Levy, P. B., J. Power and H. Thielecke, *Modelling environments in call-by-value programming languages*, Information and Computation **185** (2003), pp. 182–210.

[6] Lindley, S., P. Wadler and J. Yallop, *The Arrow Calculus*, Technical Report EDI-INF-RR-1258, University of Edinburgh (2008).

[7] Mac Lane, S., "Categories for the Working Mathematician," Number 5 in Graduate Texts in Mathematics, Springer-Verlag, 1998, 2nd edition.

[8] McBride, C. and R. Paterson, *Applicative Programming with Effects*, Journal of Functional Programming **18** (2008), pp. 1–13.

[9] Moggi, E., *Notions of computation and monads*, Information and Computation **93** (1991), pp. 55–92.

[10] Power, J., *Generic models for computational effects*, Theor. Comput. Sci. **364** (2006), pp. 254–269.

[11] Power, J. and E. Robinson, *Premonoidal categories and notions of computation*, Mathematical Structures in Computer Science **7** (1997), pp. 453–468. URL ftp://ftp.dcs.qmw.ac.uk/pub/lfp/edmundr/premoncat.dvi.gz

[12] Power, J. and H. Thielecke, *Closed Freyd- and kappa-categories*, in: *ICALP*, LNCS **1644** (1999). URL http://www.cs.bham.ac.uk/~hxt/research/freydkappa.ps

[13] Uustalu, T. and V. Vene, *Signals and Comonads*, J. UCS **11** (2005), pp. 1310–1326.

A Auxiliary Definitions and Notation

This appendix recalls several standard definitions and defines the notation we use.

A.1 Cartesian Closed Categories

The constructions of this paper are carried out over a base category C which we assume to have finite products, and sometimes assume to be cartesian closed. We work with a choice of finite product structure, and we use $x \times y$ to denote its action on objects and 1 for the terminal object or empty product. We use $\langle f, g \rangle$ to denote pairing, π_i for projections and $f \times g$ to denote the induced functor action. For cartesian closed structure we use $[x \to y]$ to denote the internal hom.

We also make use of several defined constants in categories with finite products, which have the types:

$$\begin{aligned} id : x &\to x \\ \alpha : (x \times y) \times z &\to x \times (y \times z) \\ \sigma : x \times y &\to y \times x \\ -;- : [x \to y] \times [y \to z] &\to [x \to z] \end{aligned}$$

We also use the same notation for the "internal" versions of these constants, which have type $1 \to -$.

A.2 Enriched Categories

Enriched categories generalise ordinary categories by instead of assuming a set of morphisms between objects, we require a hom-object of some base category C. The canonical reference for enriched categories is Kelly [4]. Formally, given a cartesian closed category C, a C-enriched category D consists of a collection of objects $|D|$ and for every pair $x, y \in |D|$ a C object $D(x,y)$—the hom-objects of D—and two families of C-morphisms $\underline{id}_x : 1 \to D(x,x)$ and $\underline{\ggg}_{xyz} : D(x,y) \times D(y,z) \to D(x,z)$, satisfying the diagrams:

$$\begin{tikzcd}
D(x,y) \times 1 \arrow[d, "id \times \underline{id}"'] & & D(x,y) \arrow[ll, "\cong"'] \arrow[rr, "\cong"] & & 1 \times D(x,y) \arrow[d, "id \times \underline{id}"] \\
D(x,y) \times D(y,y) \arrow[urr, "\underline{comp}"'] & & & & D(x,x) \times D(x,y) \arrow[ull, "\underline{comp}"]
\end{tikzcd}$$

$$\begin{tikzcd}
D(w,x) \times D(x,y) \times D(y,z) \arrow[r, "\underline{comp} \times id"] \arrow[d, "id \times \underline{comp}"'] & D(w,y) \times D(y,z) \arrow[d, "\underline{comp}"] \\
D(w,x) \times D(x,z) \arrow[r, "\underline{comp}"] & D(w,z)
\end{tikzcd}$$

Given two C-categories D and D', a C-functor $F : D \to D'$ is given by a mapping of D objects to D' objects and a family of C-morphisms $F_{xy} : D(x,y) \to D(Fx, Fy)$, satisfying the diagrams:

$$\begin{tikzcd}
1 \arrow[d, "id^D"'] \arrow[dr, "id^{D'}"] & \\
D(x,y) \arrow[r, "F"] & D'(Fx,Fy)
\end{tikzcd}$$

$$\begin{tikzcd}
D(x,y) \times D(y,z) \arrow[r, "\underline{comp}^D"] \arrow[d, "F \times F"'] & D(x,z) \arrow[d, "F"] \\
D'(Fx,Fy) \times D'(Fy,Fz) \arrow[r, "\underline{comp}^{D'}"] & D'(Fx,Fz)
\end{tikzcd}$$

Given two C-functors F and G from D to D', a C-natural transformation between them is a family of C-morphisms $\zeta_x : 1 \to D'(Fx, Gx)$ satisfying the diagram:

$$\begin{tikzcd}
D(x,y) \times 1 \arrow[r, "F \times \zeta_y"] & D'(Fx,Fy) \times D'(Fy,Gy) \arrow[dr, "\underline{comp}"] & \\
D(x,y) \arrow[u, "\cong"'] \arrow[d, "\cong"] & & D'(Fx,Gy) \\
1 \times D(x,y) \arrow[r, "\zeta_x \times F"] & D'(Fx,Gx) \times D'(Gx,Gy) \arrow[ur, "\underline{comp}"'] &
\end{tikzcd}$$

A.3 Indexed Categories

Given a category C, a C-indexed category is a functor H from $C^{\mathrm{op}} \to \mathrm{Cat}$, where Cat is the category of all categories and functors between them. For an object w of C we use H_w to denote the category that H gives us. The contravariant action of H turns every morphism $f : w \to w'$ in C into a re-indexing functor $H_f : H_{w'} \to H_w$. Given two C-indexed categories H and H', an identity-on-objects C-indexed functor F between them is a family of functors $F_w : H_w \to H'_w$ such that for all $f : w \to w'$ in C, $H_f(F_{w'}x) = F_w(H_f x)$ on objects and $H_f(F_{w'}g) = F_w(H_f g)$ on morphisms.

Assuming C has finite products, we can construct an indexed category $\widehat{C}$ using C's finite products. Every object w of C gives a comonad by $W_w y = w \times x$, with counit $\epsilon = \pi_2 : W_w x \to x$ and comultiplication $\delta = \langle \pi_1, id \rangle : W_w x \to W_w W_w x$. Given a morphism $f : w \to w'$ there is a morphism of comonads $W_f : W_w \to W_{w'}$ given by $f \times -$. Each comonad has an associated co-Kleisli category, which we write as $\widehat{C}_w$ which has $\widehat{C}_w(x, y) = C(W_w x, y) = C(w \times x, y)$. Morphisms $f : w \to w'$ induce functors $\widehat{C}_f : \widehat{C}_{w'} \to \widehat{C}_w$ via precomposition with the induced comonad morphisms. Hence we have an C-indexed category $\widehat{C}$, where every fibre has exactly the same objects as C. Moreover, each fibre $\widehat{C}_w$ of this indexed category has finite products, a choice of which can be defined in terms of our chosen finite products on C. It is also easy to see that $C \cong \widehat{C}_1$.