Category Theory with Strings

Shunsuke Sogame

March 21, 2016

1 Introduction

This is a complementary document for introductory books of category theory ([1], [2], [3], [8]) using *string diagrams*. Don't trust my poor mathematics. Any feedback is welcome at github.com/okomok/strcat.

2 Preliminaries

2.1 Universality

Definition 2.1 For a boolean-valued function P, define

$$!aP(a) \coloneqq P(a) \wedge \forall a'(P(a') \implies a = a')$$

Definition 2.2 (Uniqueness Quantification) Define

$$\exists!aP(a) \coloneqq \exists a!aP(a)$$

meaning that "there exists a unique a such that P".

Remark 2.3 On the other hand,

$$\exists a\big((!aP(a)) \wedge Q(a)\big)$$

means "there exists a unique a such that P, furthermore the a is Q".

Definition 2.4 (Universality) Given a binary boolean-valued function P, we boldly call a statement of the form

$$(\forall x \in X)(\exists!y \in Y)(P(x,y))$$

the *universality* of P.

Proposition 2.5 (Functional Universality)

$$
\begin{aligned}
&(\forall x \in X)(\exists!y \in Y)(P(x,y)) \\
\iff &(\exists f : X \to Y)(\forall x \in X)(\forall y \in Y)(P(x,y) \iff y = f(x))
\end{aligned}
$$

PROOF. ($\implies$) by the Axiom of choice. ($\impliedby$) immediate. $\qquad\square$

Definition 2.6 (Functional Bijectivity) Given a a function $g : Y \to X$, a statement

$$(\exists f : X \to Y)(\forall x \in X)(\forall y \in Y)(x = g(y) \iff y = f(x))$$

is known as the *bijectivity* of f and g. This is a special case of universality where $P(x, y)$ is $x = g(y)$.

2.2   Lambda Expressions

Definition 2.7 (Lambda Expression) Following famous symbols like Σ, define

$$\Lambda_x y := x \mapsto y$$

for anonymous functions.

Definition 2.8 Given a function H whose domain is a set of functions, define

$$H_x y := H(\Lambda_x y)$$

Definition 2.9 (Placeholder Expression) For simple lambda expressions, you may use *placeholders*:

$$?{+}1 := \Lambda_n n{+}1$$

Placeholder symbols can vary: `?`, `-`, `1`, etc.

2.3   Families

Syntax of the function application is world-standard and fixed:

$$f(x) \text{ or } fx$$

but sometimes you might want cuter syntax like that

$$\widehat{x}$$

Definition 2.10 (Family) A *family declaration* is a way to provide a function with arbitary application syntax. The usage is clear from an example

$$(\widehat{x} \in Y)_{x \in X}$$

We call such a function a *family*. Furthermore, a function body can be placed like that

$$(\widehat{x} := x^2 \in Y)_{x \in X}$$

Example 2.11 The most-used family declaration is the subscript style $(a_i)_i$. You can view a tuple $(a_1, a_2, \ldots, a_n)$ to be an abbreviation of $(a_i)_{i \in \{1,2,\ldots,n\}}$.

Families can do more.

Definition 2.12 (Dependent Function) Let F a set-valued function.

$$(f(x) \in F(x))_{x \in X}$$

defines a function

$$f : X \to \bigcup_{x \in X} F(x)$$

such that

$$(\forall x \in X)(f(x) \in F(x))$$

Such f is called a *dependent function*, for the $F(x)$ depends on x. In case F is a constant function, f is a normal function $X \to Y$.

2.4   Coherence

It is sure you write

$$3 + 1 + 2$$

rather than

$$(3 + (0 + 1)) + 2$$

because you know the arithmetic laws

$$x + (y + z) = (x + y) + z$$
$$0 + x = x = x + 0$$

disambiguate unparenthesized expressions. Informally laws to introduce simpler syntax are called *coherence conditions* or briefly *coherence*.

3 Categories

3.1 The Definition

Definition 3.1 (Category) A *category* $\mathcal{C}$ consists of

1. *objects*: a class $\mathrm{Ob}(\mathcal{C})$

2. *morphisms* or *hom-sets*: a family of sets $(\mathcal{C}(A, B))_{A,B \in \mathrm{Ob}(\mathcal{C})}$

3. *compositions*: a family of functions

$$(\circ : \mathcal{C}(B, C) \times \mathcal{C}(A, B) \to \mathcal{C}(A, C))_{A,B,C \in \mathrm{Ob}(\mathcal{C})}$$

4. *idenitities* or *units*: a family of morphisms

$$(\mathrm{id}_A \in \mathcal{C}(A, A))_{A \in \mathrm{Ob}(\mathcal{C})}$$

satisfying the following coherence conditions

1. *associativity*: for any $f \in \mathcal{C}(A, B)$, $g \in \mathcal{C}(B, C)$, and $h \in \mathcal{C}(C, D)$,

$$h \circ (g \circ f) = (h \circ g) \circ f$$

2. *unitality*: for any $f \in \mathcal{C}(A, B)$,

$$\mathrm{id}_B \circ f = f = f \circ \mathrm{id}_A$$

A morphism $f \in \mathcal{C}(A, B)$ is often denoted as $f : A \to B$.

3.2 String Diagrams

From now on, we will introduce *string diagrams* to complement(or hopefully replace) commutative diagrams, where an object A is depicted as an optionally-tagged string

A morphism $f \in \mathcal{C}(A, B)$ is depicted as a node

A composition joins two strings:

$$\begin{array}{c} C \\ \boxed{g} \\ B \\ \boxed{f} \\ A \end{array} := g \circ f$$

An idenity is indistinguishable from an object:

$$\Big|_A := \mathrm{id}_A$$

Check these diagrams create no ambiguity thanks to the coherence.

Definition 3.2 (Isomorphism) An *isomorphism* is a pair of morphisms

$$f : A \to B$$
$$g : B \to A$$

satisfying the *invertibility*

$$\begin{array}{c} A \\ \boxed{g} \\ B \\ \boxed{f} \\ A \end{array} = \Big|_A \quad \text{and} \quad \begin{array}{c} B \\ \boxed{f} \\ A \\ \boxed{g} \\ B \end{array} = \Big|_B$$

Definition 3.3 (Functional Box) Given categories $\mathcal{C}$ and $\mathcal{C}'$, a function

$$h : \mathcal{C}(A, B) \to \mathcal{C}'(A', B')$$

is depicted as a box

$$h \;\; \boxed{?}$$

Definition 3.4 (Opposite Category) Given a category $\mathcal{C}$ and a morphism

$$\begin{array}{c} B \\ \boxed{f} \\ A \end{array}$$

you can build a category with strings upsidedown:

$$\begin{array}{c} \big| A \\ \boxed{f} \\ \big| B \end{array}$$

which is denoted as $\mathcal{C}^{\mathrm{op}}$ the *opposite category* of $\mathcal{C}$.

Definition 3.5 (Discrete Category) A category $\mathcal{C}$ such that

$$A = B \implies \mathcal{C}(A, B) = \{\mathrm{id}_A\}$$
$$A \neq B \implies \mathcal{C}(A, B) = \emptyset$$

is called a *discrete category*. Any set can be represented as a discrete category.

Definition 3.6 (Product Category) Given two categories $\mathcal{A}$ and $\mathcal{B}$, the *product category*

$$\mathcal{A} \times \mathcal{B}$$

is depicted as parallel strings

$$\begin{array}{cc} \mathcal{A} & \mathcal{B} \\ \big| A' & \big| B' \\ \boxed{a} & \boxed{b} \\ \big| A & \big| B \end{array}$$
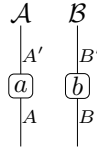
A composition, which joins parallel strings, is defined by
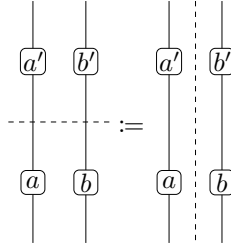
An identity is trivially

By these definitions,
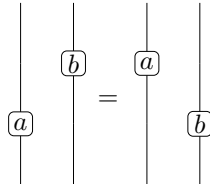
4 Functors

4.1 The Definition

Definition 4.1 (Functor) A *functor* $F : \mathcal{C} \to \mathcal{D}$ consists of:

1. *domain*: a category $\mathcal{C}$

2. *codomain*: a category $\mathcal{D}$

3. a family of objects $(FA \in \mathrm{Ob}(\mathcal{D}))_{A \in \mathrm{Ob}(\mathcal{C})}$

4. families of morphisms

$$\Big((F(f) \in \mathcal{D}(FA, FB))_{f \in \mathcal{C}(A,B)}\Big)_{A,B \in \mathrm{Ob}(\mathcal{C})}$$

satisfying the *functoriality*:

1. *composition-compatibility*: for any $f \in \mathcal{C}(A, B)$ and $g \in \mathcal{C}(B, C)$,

$$F(g \circ f) = F(g) \circ F(f)$$

2. *unit-compatibility*: for any $A \in \mathrm{Ob}(\mathcal{C})$,

$$F(\mathrm{id}_A) = \mathrm{id}_{FA}$$

Definition 4.2 (Infrafunctor) An *infrafunctor* is a functor without the requirement of functoriality.

4.2 Functorial Tubes

In string diagrams, a functor is represented as a tube

Placeholders make it simple:

One can check the functoriality ensures any tube like

$$\begin{bmatrix} & C \\ & \boxed{g} \\ & B \\ & \boxed{f} \\ & A \end{bmatrix}$$
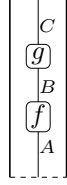
be unambiguous. "Join then tube" is the same as "Tube then join".

Proposition 4.3 Any functor preserves isomorphisms meaning that

$$(\begin{matrix} B \\ \boxed{f} \\ A \end{matrix} , \begin{matrix} A \\ \boxed{g} \\ B \end{matrix}) : \text{isomorphism} \implies (\begin{bmatrix} B \\ \boxed{f} \\ A \end{bmatrix} , \begin{bmatrix} A \\ \boxed{g} \\ B \end{bmatrix}) : \text{isomorphism}$$

PROOF. Immediate by functoriality that inheres in tubes. □

Definition 4.4 (Composite Functor) For any two functors

$$F : \mathcal{A} \to \mathcal{B}$$
$$G : \mathcal{B} \to \mathcal{C}$$

, the *composite functor* of F and G

$$G \circ F : \mathcal{A} \to \mathcal{C}$$

is depicted as

$$\begin{bmatrix} & & \\ G & F & \boxed{?} \\ & & \end{bmatrix}$$
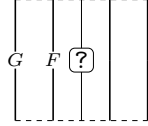
Definition 4.5 (Identity Functor) An *idenity functor*

$$\mathrm{Id}_{\mathcal{C}} : \mathcal{C} \to \mathcal{C}$$

is depicted as

$$\begin{bmatrix} \mathrm{Id} \; \boxed{?} \end{bmatrix} := \boxed{?}$$
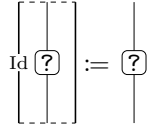
Definition 4.6 (Contravariant Functor) A functor whose domain is an opposite category

$$F : \mathcal{C}^{\mathrm{op}} \to \mathcal{D}$$

is called *contravariant*, while a normal functor is called *covariant*.

A contravariant functor is depicted as
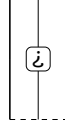
Definition 4.7 (Variant) Given a statement regarding functors, you can obtain a corresponding one regarding contravariant functors and vice versa. We call such a statement the *variant* of the original one.

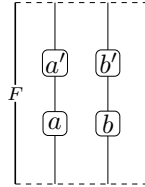**Definition 4.8 (Binary Functor)** A functor whose domain is a product category

$$F : \mathcal{A} \times \mathcal{B} \to \mathcal{C}$$

is called a *binary functor* or *bifunctor*.

With numbered placeholders, it is depicted as
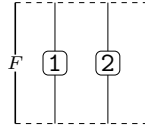
Spelling out the definition of functoriality, one can check a diagram like

is unambiguous.

Definition 4.9 (Partial Application) Given a binary functor $F : \mathcal{A} \times \mathcal{B} \to \mathcal{C}$, a *partially applied* functor

$$\Lambda_B F(A, B) : \mathcal{B} \to \mathcal{C} \text{ or briefly}$$
$$F(A, ?) : \mathcal{B} \to \mathcal{C}$$

is defined by
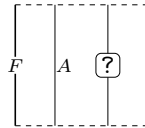
The definition of $F(?, B)$ is an exercise.

Definition 4.10 (Small Category) A category $\mathcal{C}$ is called *small* when its $\mathrm{Ob}(\mathcal{C})$ is a set.

Definition 4.11 (Category of Small Categories) The *category of small categories* **Cat** is the category whose objects are all small categories and whose morphisms are functors:
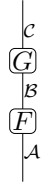
, where composite functors join the strings.

Definition 4.12 (Full and Faithful Functor) A functor $F : \mathcal{C} \to \mathcal{D}$ is called *full and faithful* if for each object A and B in $\mathcal{C}$, the family

$$(F(f) : FA \to FB)_{f:A \to B}$$

is bijective.

In other words, there is a functional box such that
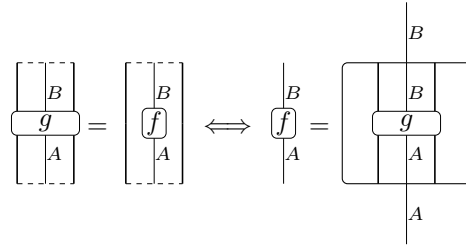
One can make the box better-looking
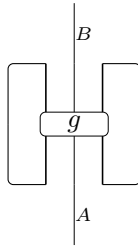
Proposition 4.13 This box has a functoriality-like property:
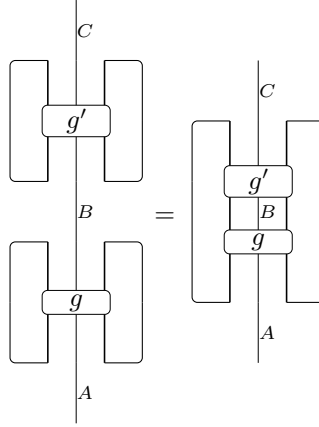
Combined with proposition 4.3,

Proposition 4.14
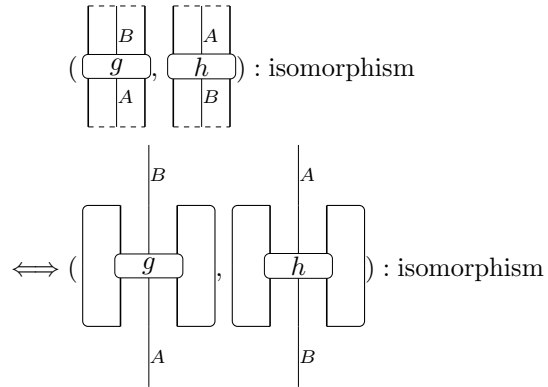
5 Natural Transformations

5.1 The Definition

Definition 5.1 (Naturality) Given two infrafunctors

$$F, G : \mathcal{C} \to \mathcal{D}$$

a family of morphisms

$$(\tau_A \in \mathcal{D}(FA, GA))_{A \in \mathrm{Ob}(\mathcal{C})}$$

is called *natural* when for any $f \in \mathcal{C}(A, B)$,

$$\tau_B \circ F(f) = G(f) \circ \tau_A$$

In case parenthese are cumbersome, you can say "τ_A is *natural in A*".

Definition 5.2 (Natural Transformation) Furthermore, in particular case F and G are functorial(then they are functors), τ is denoted as a *natural transformation*

$$\tau : F \to G$$

Remark 5.3 The orthogonality of functoriality and naturality is sometimes helpful.

5.2 Natural Connectors

In string diagrams, a natural transformation is a connector of two tubes
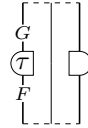
because the naturality states a node can travel between tubes

This inspires you to assign

12
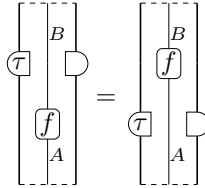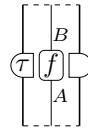
Definition 5.4 (Vertical Composition) Given three functors

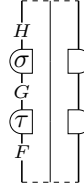$$F, G, H : \mathcal{C} \to \mathcal{D}$$

and two natural transformations

$$\tau : F \to G$$
$$\sigma : G \to H$$

the *vertical composition* of τ and σ

$$\sigma \circ \tau : F \to H$$

is defined by

Definition 5.5 (Horizontal Composition) Given four functors

$$F, G : \mathcal{A} \to \mathcal{B}$$
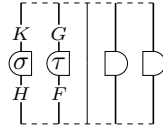$$H, K : \mathcal{B} \to \mathcal{C}$$

and two natural transformations

$$\tau : F \to G$$
$$\sigma : H \to K$$

the *horizontal composition* of τ and σ

$$\sigma\tau : H \circ F \to K \circ G$$

is defined by

You can easily check the naturality. Travel by car ferry.

Definition 5.6 (Identity Natural Transformation) Given a functor

$$F : \mathcal{C} \to \mathcal{D}$$

the *identity natural transformation*

$$\mathrm{id}_F : F \to F$$

is defined by

$$\boxed{\mathrm{id}} \; \Big\rangle := \Big[\; \Big| \; \Big| \; \Big]$$

Definition 5.7 (Whiskering) A *whikering* is a horizontal composition with identity natural transformations:
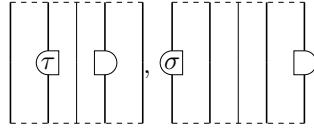
Definition 5.8 (Natural Isomorphism) A *natural isomorphism* is a pair of natural transformations

$$\tau : F \to G$$
$$\sigma : G \to F$$

satisfying the *invertibilty*:

The same symbol is often used for the pair.

Proposition 5.9 For any natural transformation τ,

$$(\forall A)(\tau_A : \text{invertible})$$

is enough to build the other natural σ.

14

Definition 5.10 (Functor Category) Given a small category $\mathcal{C}$ and a category $\mathcal{D}$, the functor category $[\mathcal{C}, \mathcal{D}]$ is a category whose objects are functors from $\mathcal{C}$ to $\mathcal{D}$ and whose morphisms are natural transformations:

$$
\begin{array}{c}
\Big|\, H \\
\boxed{\sigma} \\
\Big|\, G \\
\boxed{\tau} \\
\Big|\, F
\end{array}
$$

, where vertical compositions join the strings.

Definition 5.11 For the later use, define a lambda-tasted notation for a set of natural transformations:

$$\mathrm{Nat}_A(FA, GA) := \mathrm{Nat}(F, G) := [\mathcal{C}, \mathcal{D}](F, G)$$

6 Category of Sets

6.1 The Definition

Definition 6.1 (Category of Sets) The *category of sets* **Set** is a category whose objects are sets and whose morphisms are functions:

$$\begin{array}{c} | Z \\ \boxed{g} \\ | Y \\ \boxed{f} \\ | X \end{array}$$

, where nodes are joined by the function composition.

A category is essentially one-dimensional so far: the vertical composition only. Here we introduce the horizontal composition for functions.

Definition 6.2 (Monoidal Category of Sets) Parallel strings are defined by

$$\Big| X \quad \Big| X' := \Big| X \times X'$$

The horizontal composition is defined by

$$\begin{array}{cc} | Y & | Y' \\ \boxed{f} & \boxed{f'} \\ | X & | X' \end{array} := \Lambda_{x,x'}(f(x), f'(x'))$$

Strings for the singleton set $\{*\}$ is omitted so that an element of a set is represented as

$$\begin{array}{c} | X \\ \boxed{x} \end{array}$$

One can check any string diagram built upon these definitions is unambiguous thanks to the trivial bijections:

$$X \times (X' \times X'') \cong (X \times X') \times X''$$
$$X \times \{*\} \cong X$$

Informally such two-dimensional diagrams are called *monoidal*.

6.2 Hom-set Bands

Given a category $\mathcal{C}$, a special string, a *band*, is introduced for hom-sets:

$$\boxed{B \quad A} := \Big| \mathcal{C}(A,B)$$

16

A space-saving form is depicted as

$$B \parallel A$$

Remark 6.3 Note that the order of objects is flipped. This is resulting from the unfortunate convention that we write $b = h(a)$ but not $h : B \leftarrow A$.

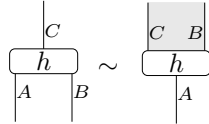The composition of morphisms can be depicted as

 or
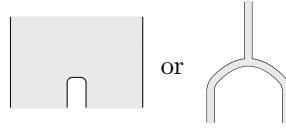
Identity morphisms can be depicted as

 or

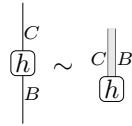As an exercise, write down the associativity and unitality using these diagrams.

Definition 6.4 (Currying) In particular case $\mathcal{C} = \mathbf{Set}$, there exists the *curry bijection*

$$\mathbf{Set}(A \times B, C) \cong \mathbf{Set}(A, \mathbf{Set}(B, C))$$

We don't distinguish the two diagrams, for the naturality of the bijection ensures "Move the right-side leg up and down" works correct.

Definition 6.5 (Naming) In case A is the singleton set,

is called a *naming*, which turns a function to an element of function-sets.

Definition 6.6 (Hom-Functor) Hom-sets can be extended to a binary functor

$$\Lambda_{A,B} \mathcal{C}(A, B) : \mathcal{C}^{\mathrm{op}} \times \mathcal{C} \to \mathbf{Set} \text{ or briefly}$$

17

$$\mathcal{C}(-,+) : \mathcal{C}^{\mathrm{op}} \times \mathcal{C} \to \mathbf{Set} \text{ or briefly}$$
$$\mathrm{Hom}_{\mathcal{C}} : \mathcal{C}^{\mathrm{op}} \times \mathcal{C} \to \mathbf{Set}$$

defined by
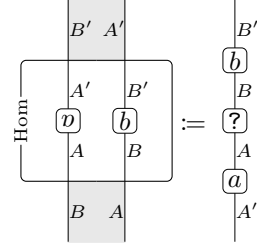
where the world in the box is the product category $\mathcal{C}^{\mathrm{op}} \times \mathcal{C}$.

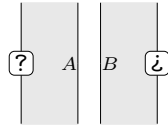This definition will inspire you to depict the hom-functors as
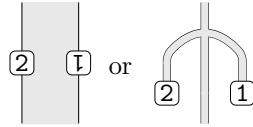
that looks topologically equivalent.

Definition 6.7 (Unary Hom-Functor) According to definition 4.9,

$$\mathcal{C}(A,+) : \mathcal{C} \to \mathbf{Set}$$
$$\mathcal{C}(-,B) : \mathcal{C}^{\mathrm{op}} \to \mathbf{Set}.$$

are respectively depicted as

7 The Yoneda Lemma

Definition 7.1 Given a functor $F : \mathcal{C}^{\mathrm{op}} \to \mathbf{Set}$ and an object A in $\mathcal{C}$, a natural tranformation of the form

$$(\tau_X : \mathcal{C}(X, A) \to FX)_X$$

can be depicted as
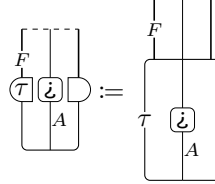
owing to the naturality.

Definition 7.2 (Yoneda Bijection) The *Yoneda bijection* is defined as
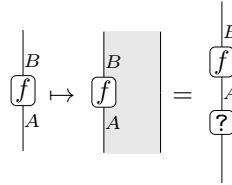
$$\mathrm{Nat}_X(\mathcal{C}(X, A), FX) \cong FA$$
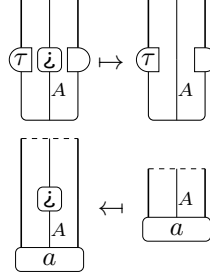
Lemma 7.3 (Yoneda Lemma) The Yoneda bijection is actually bijective and natural in F and A.

PROOF. Now the proof is on my soul trivial! □

Definition 7.4 (Yoneda Embedding) The *Yoneda embedding* is defined as

$$\Lambda_A \Lambda_X \mathcal{C}(X, A) : \mathcal{C} \to [\mathcal{C}^{\mathrm{op}}, \mathbf{Set}]$$

using the diagram of hom functors. In short,

Definition 7.5 A natural transformation of the form

$$(\tau_X : \mathcal{C}(X, A) \to \mathcal{C}(X, B))_X$$

can be depicted as
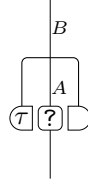
Definition 7.6 (Yoneda Embedding Bijection) In special case $F := \mathcal{C}(\text{-}, B)$, the Yoneda bijection is expanded to

$$\mathrm{Nat}_X(\mathcal{C}(X, A), \mathcal{C}(X, B)) \cong \mathcal{C}(A, B)$$
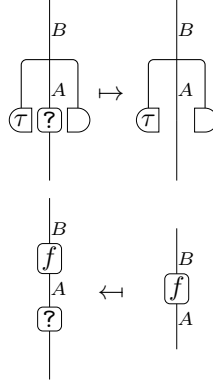
You will notice the second mapping is the Yoneda embedding so that it is full and faithful. Combined with proposition 4.14,

Proposition 7.7 (Yoneda Principle)

20

8 Representations

Definition 8.1 (Representation) Given a functor $H : \mathcal{C} \to \textbf{Set}$, a *representation* of H is a pair of

1. an object R in $\mathcal{C}$

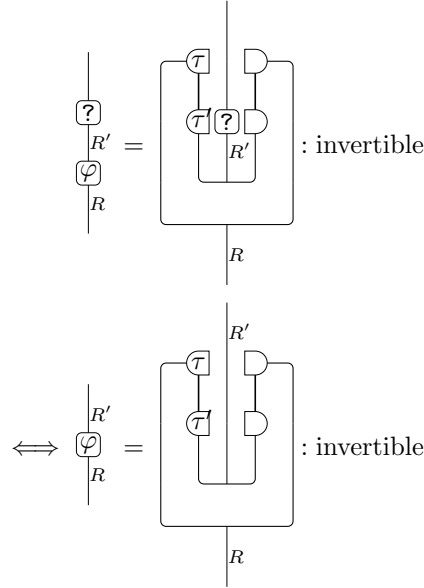2. a natural bijection $(\tau_X : HX \cong \mathcal{C}(R, X))_X$

The bijectivity can be expressed using weird boxes
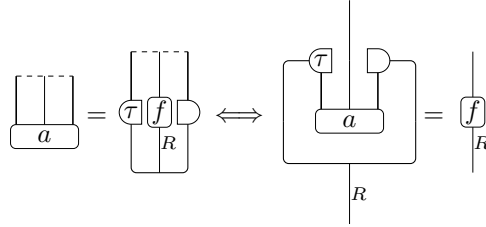
thanks to the naturality. The following proposition allows us to call a representation *the* representaiton denote as *textpreH*.

Proposition 8.2 (Uniqueness of Representations) Representations are unique up to unique isomorphism.

PROOF. Let (R', τ') be another representation. By the variant of proposition 7.7,

$\Box$

Definition 8.3 Given a functor $H : \mathcal{B}^{\mathrm{op}} \times \mathcal{A} \to \mathbf{Set}$, a natural bijection of the form

$$(\tau_X : H(B, X) \cong \mathcal{A}(A, X))_X$$

can be expressed by
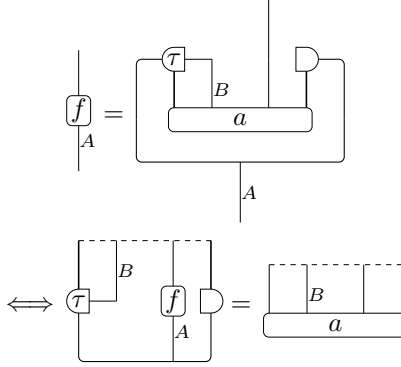
Proposition 8.4 (Parameterized Representations) Let $H : \mathcal{B}^{\mathrm{op}} \times \mathcal{A} \to \mathbf{Set}$ be a functor. Given a family of objects $(SB)_B$ and a family of representations
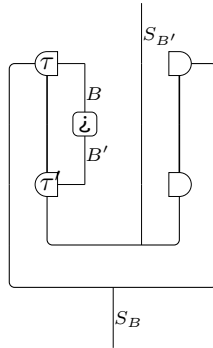
$$((\tau_X^B : H(B, X) \cong \mathcal{A}(SB, X))_X)_B$$

there exists a unique family

$$(S(f) \in \mathcal{A}(SB, SB'))_{f \in \mathcal{B}(B, B')}$$

such that τ is natural in B. Furthermore, S is functorial.

PROOF. Define S as

9   Limits

Definition 9.1 (Cone) Given a functor $F : \mathcal{A} \to \mathcal{B}$, a cone of F consists of

1. an object B in $\mathcal{B}$

2. a natural transformation $(v_X : B \to FX)_X$

Definition 9.2 (Conicality) We may call the naturality of a cone explicitly the *conicality*, which can be expressed as

like a magical box any morphism can appear from.

Remark 9.3 Vertical and horizontal composition preserve conicality, a special case of naturality.

Definition 9.4 (Limit) Given a functor $F : \mathcal{A} \to \mathcal{B}$, a limit of F is a pair of

1. an object in $\mathcal{B}$ denoted as $\lim F$

2. a natural bijection $(\mathcal{B}(B, \lim F) \cong \mathrm{Nat}_X(B, FX))_B$

Definition 9.5 (Limiting Cone) The limit bijectivity, thanks to its naturality, can be expressed as

where $\boxed{\lim}$ is a cone called a *limiting cone* of F.

The following proposition allows us to call a limit *the* limit.

Proposition 9.6 Limits are unique up to isomorphism.

PROOF. Immediate by proposition 8.2, because a limit is nothing but a contravariant representation

$$\mathrm{rep}_B \, \mathrm{Nat}_X(B, FX)$$

$\sqcup$

Proposition 9.7 A limiting cone is *monic* meaning that

PROOF. Immediate by the limit bijectivity. ⊓

Definition 9.8 (Product) In particular case the domain of a functor $F : \mathcal{A} \to \mathcal{B}$ is discrete, the limit of F is called the *product* of F denoted as $\prod F$.

Definition 9.9 (Projection) Spelling out the product bijectivity,

where $\boxed{\pi}$ is called the *projection* of F.

Remark 9.10 Conicality has no concern here, because any family of the form

$$(v_X : B \to FX)_{X \in \mathrm{Ob}(\mathcal{A})}$$

is always natural in case $\mathcal{A}$ is discrete.

Example 9.11 In case F is a functor $X \to \mathcal{S}et$ with a set X(as a discrete category), the product of F is a set of dependent functions

$$\textstyle\prod_x F(x) \cong \{f \mid (f(x) \in F(x))_x\}$$

Definition 9.12 (Dual) Given a statement containing string diagrams, by flipping it upside down, a corresponding statement is obtained. It is called the *dual* of the original one.

Definition 9.13 (Coproduct) A *coproduct* is a structure obtained from the bijectivity diagram of products flipped.

24

Remark 9.14 The dual makes a codomain opposite, while the variant does for a domain.

Definition 9.15 (Preservation of Limits) Given a functor $F : \mathcal{A} \to \mathcal{B}$ and a limiting cone of F
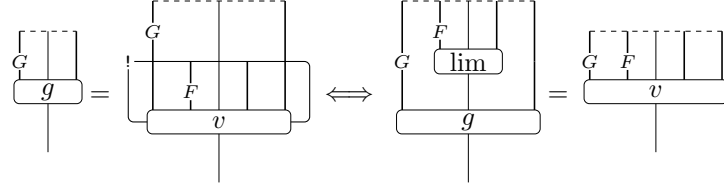
$$(\lim_X : \lim F \to FX)_X$$

a functor $G : \mathcal{B} \to \mathcal{C}$ *preserves limits* of F when

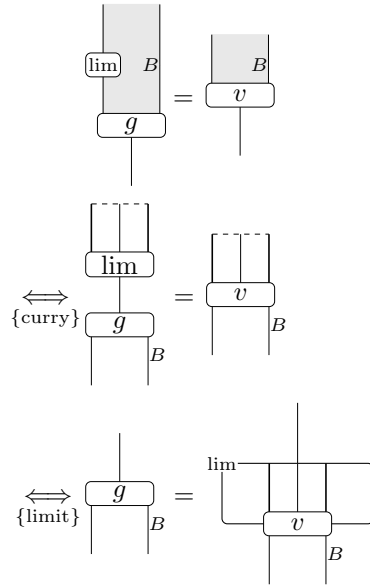$$(G(\lim_X) : G\lim F \to GFX)_X$$
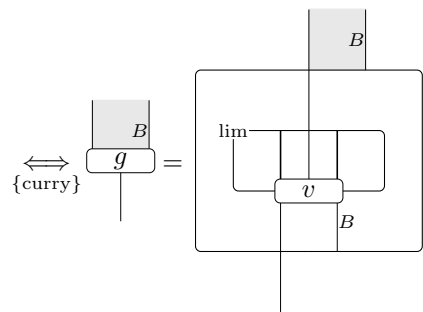
is a limiting cone of $G \circ F$.

In diagrams, G is such that there exists some box ! satisfying

Proposition 9.16 (HFPL) Hom-functors preserve limits, meaning that given a functor $F : \mathcal{A} \to \mathcal{B}$ and an object B in $\mathcal{B}$, the covariant hom-functor $\mathcal{B}(B, \mathbin{+}) : \mathcal{B} \to \mathbf{Set}$ preserves limits of F.

PROOF.

25

$\overset{\Longleftrightarrow}{\{\text{curry}\}}$

B

g

$=$

B

lim

v

B

$\Box$

10 Adjunctions

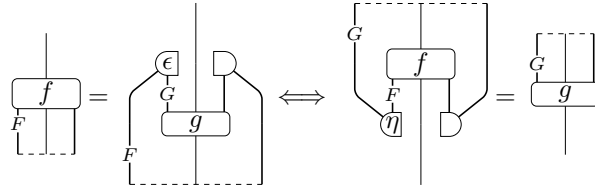**Definition 10.1 (Adjunction)** Given two categories $\mathcal{C}$ and $\mathcal{D}$, an *adjunction*
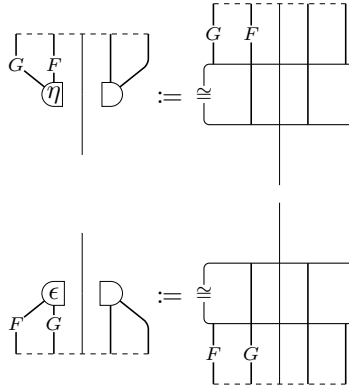
$$F \dashv G$$

consists of

1. *left adjoint*: a functor $F : \mathcal{C} \to \mathcal{D}$

2. *right adjoint*: a functor $G : \mathcal{D} \to \mathcal{C}$

3. *adjunct*: a natural bijection

$$(\mathcal{D}(FC, D) \cong \mathcal{C}(C, GD))_{C,D}$$

A nice consequence is that this bijectivity needs no boxes, expressed by natural transformations only.

where

called respectively the *unit* and *counit*.

Proposition 10.2 Given a functor $G : \mathcal{D} \to \mathcal{C}$, a family of natural bijections
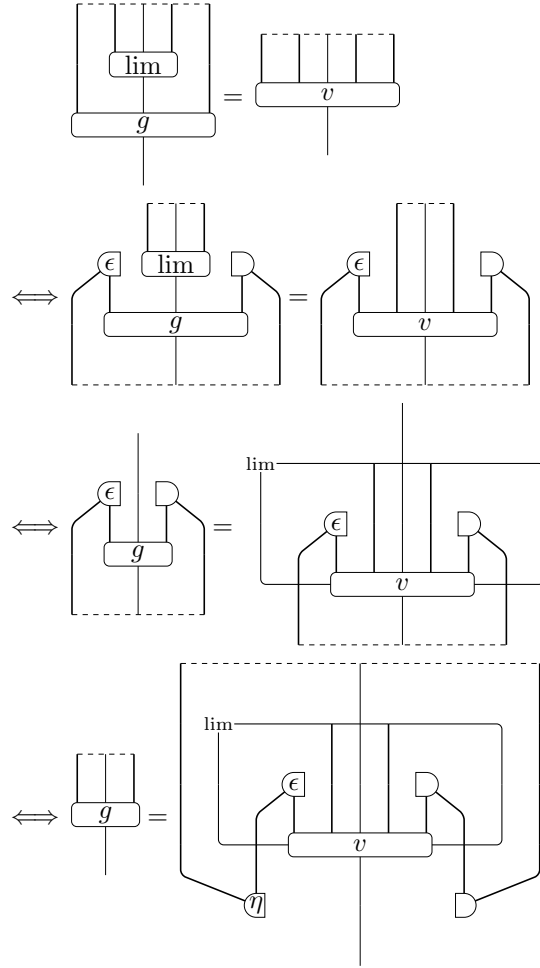
$$((\mathcal{C}(C, GD) \cong \mathcal{D}(F_c, D))_D)_C$$

is enough to construct the adjunction $F \dashv G$.

PROOF. Immediate by $H(C, D) \coloneqq \mathcal{C}(C, GD)$ in proposition 8.4. $\qquad\qquad$ ⌑

Proposition 10.3 (RAPL) Right adjoints preserve limits, meaning that given an adjunction $F \dashv (G : \mathcal{D} \to \mathcal{C})$ and a functor $T : \mathcal{B} \to \mathcal{D}$,

$$(\lim_X : \lim T \to TX)_X : \text{limiting cone}$$
$$\Longrightarrow (G(\lim_X) : G\lim T \to GTX)_X : \text{limiting cone}$$
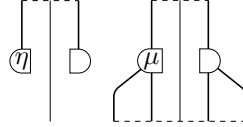
PROOF.

11 Monads

11.1 The Definition

Definition 11.1 (Monad) Given a category $\mathcal{C}$, a *monad* on $\mathcal{C}$ consists of

1. a functor $T : \mathcal{C} \to \mathcal{C}$

2. *unit*: a natural transformation $\eta : \mathrm{Id}_T \to T$

3. *multiplication*: a natural transformation $\mu : T \circ T \to T$
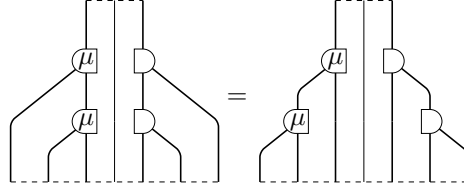
satisfying the coherence conditions

1. *associativity*: $\mu \circ T\mu = \mu \circ \mu T$

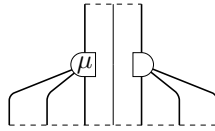2. *unitality*: $\mu \circ T\eta = \mathrm{Id}_T = \mu \circ \eta T$

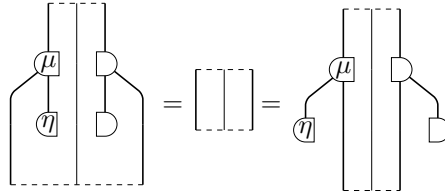A unit and multiplication are depicted respectively as

The associativity is depicted as

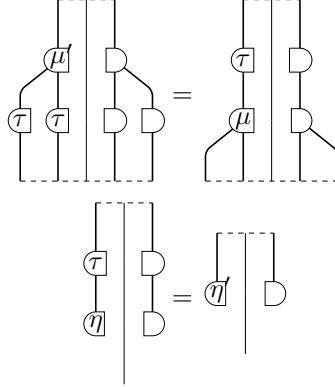This inspires you to assign

The unitality is

Definition 11.2 (Monad Morphism) Given a category $\mathcal{C}$, a *monad morphism* consists of

1. *domain*: a monad (T, η, μ) on $\mathcal{C}$

2. *codomain*: a monad (T', η', μ') on $\mathcal{C}$

3. a natural transformation $\tau : T \to T'$

satisfying the coherence conditions

1. *multiplication-compatibility*: $\tau \circ \eta = \eta'$

2. *unit-compatibility*: $\tau \circ \mu = \mu' \circ \tau\tau$

The coherence is depicated as

Definition 11.3 (Categoy of Monads) Given a category $\mathcal{C}$, the *category of monads* $\mathbf{Mnd}(\mathcal{C})$ is a category whose objects are monads and whose morphisms are monad morphisms.
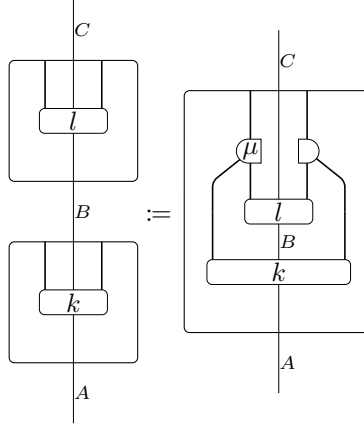
11.2   Kleisli Categories

Definition 11.4 (Kleisli Category) Given a monad (T, η, μ) on $\mathcal{C}$, the *Kleisli category* of T, denoted as $\mathcal{C}_T$ is a category consisting of

1. $\mathrm{Ob}(\mathcal{C}_T) \coloneqq \mathrm{Ob}(\mathcal{C})$

2. $\mathcal{C}_T(A, B) \coloneqq \mathcal{C}(A, TB)$

3. $l \circ k \coloneqq \mu \circ T(l) \circ k$
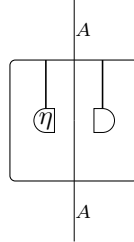
4. $\mathrm{id}_A \coloneqq \eta_A$

In diagrams, a morphism in $\mathcal{C}_T$ is depicted as a *Kleisli box*

$$\mathcal{C}_T$$
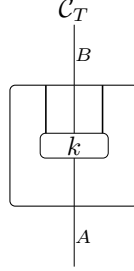
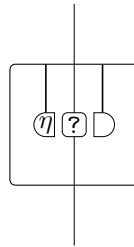

The composition is defined as
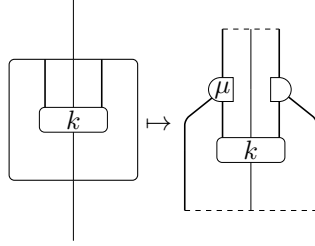

An identity morphism is defined as

Definition 11.5 (Kleisli Adjunction) Define a functor $L : \mathcal{C} \to \mathcal{C}_T$ as

$K : \mathcal{C}_T \to \mathcal{C}$ as

then they consitute the *Kleisli adjunction* $L \dashv K$ whose adjunct is the Kleisli boxing.

References

[1] Jiri Adamek, Horst Herrlich, and George E Strecker. *Abstract and Concrete Categories: The Joy of Cats (Dover Books on Mathematics)*. Dover Publications, 8 2009.

[2] Steve Awodey. *Category Theory (Oxford Logic Guides)*. Oxford University Press, 2 edition, 8 2010.

[3] Francis Borceux. *Handbook of Categorical Algebra: Volume 1, Basic Category Theory (Encyclopedia of Mathematics and its Applications)*. Cambridge University Press, 1 edition, 4 2008.

[4] John Bourke and Micah Blake McCurdy. Frobenius morphisms of bicategories, 2009.

[5] Ralf Hinze. Kan extensions for program optimisation or: Art and dan explain an old trick. In *Mathematics of Program Construction*, pages 324–362. Springer, 2012.

[6] Max Kelly. *Basic Concepts of Enriched Category Theory (London Mathematical Society Lecture Note Series)*. Cambridge University Press, 4 1982.

[7] Aleks Kissinger. Pictures of processes: automated graph rewriting for monoidal categories and applications to quantum computing. *arXiv preprint arXiv:1203.0202*, 2012.

[8] Saunders Mac Lane. *Categories for the Working Mathematician (Graduate Texts in Mathematics)*. Springer, 2nd ed. 1978. softcover reprint of the original 2nd ed. 1978 edition, 11 2010.

[9] Dan Marsden. Category theory using string diagrams. *CoRR*, abs/1401.7220, 2014.

[10] Micah Blake McCurdy. Strings and stripes, graphical calculus for monoidal functors and monads, 2010.

[11] Paul-André Mellies. Functorial boxes in string diagrams. In *Computer science logic*, pages 1–30. Springer, 2006.

[12] Peter Selinger. A survey of graphical languages for monoidal categories. In *New structures for physics*, pages 289–355. Springer, 2010.

[13] Daniele Turi. Category theory lecture notes. *Laboratory for Foundations of Computer Science, University of Edinburgh*, 2001.