Category theory: a programming language-oriented introduction

Pierre-Louis Curien

October 19, 2008

Chapter 1

Categories, functors, natural transformations

In this chapter, we introduce the basic material of category theory. In Section 1.1, we introduce the notion of category, we single out remarkable classes of morphisms, and we define the important notion of cartesian closed category. In Section 1.2, we define the notion of limit and colimit, and we spell out remarkable special casess: equalisers and coequalisers, pullbacks and pushouts. In Section 1.3, we define functors, which are morphisms between categories, and natural transformations, which are morphisms between functors. In Section 1.4, we study a special class of functors, the presheaves. In Section 1.5, we introduce pairs of adjoint functors, which are in abundance. In Section 1.6, we define the notion of equivalence of categories (a special case of adjunction). Finally, in Section 1.7 we define monads, a notion related to adjunction.

1.1 Categories

Definition 1.1.1 *A category* $\mathbf{C}$ *is given by the following data:*

- *a collection* $Obj(\mathbf{C})$ *of objects* $A, B, \ldots$ *(we write* $A : \mathbf{C}, B : \mathbf{C}, \ldots$*),*
- *a family of collections* $\mathbf{C}[A, B]$ *doubly indexed over* $Obj(\mathbf{C})$ *whose elements* f *are called morphisms (notation* $f : A \to B$*),*
- *a family of morphisms* $id_A : A \to A$ *indexed over* $Obj(\mathbf{C})$*, and*
- *a family of composition operations: if* $f : A \to B$ *and* $g : B \to C$*, then* $g \circ f : A \to C$*, such that the following equations are universally satisfied:*

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

We also write gf *for* $g \circ f$*.*

The above equalities are understood only when the compositions are well-defined (in particular, id stands for some appropriate id_A). This is spelled out in the following "proof-theoretical" presentation:

$$\frac{}{id_A : A \to A} \qquad \frac{f : A \to B \quad g : B \to C}{g \circ f : A \to C}$$

$$\frac{f : B \to A \;\; g : C \to B \;\; h : D \to C}{(f \circ g) \circ h = f \circ (g \circ h) : D \to A} \quad \frac{f : A \to B}{f \circ id_A = f : A \to B} \quad \frac{f : A \to B}{id_B \circ f = f : A \to B}$$

which stresses the fact that a category is a "typed monoid".

An alternative definition of a category is by means of two collections $Obj(\mathbf{C})$ and $Mor(\mathbf{C})$, where $Mor(\mathbf{C})$ stands for the disjoint union of all the $\mathbf{C}[A, B]$'s. In this presentation, one must also provide two functions *dom* and *cod* from $Mor(\mathbf{C})$ to $Obj(\mathbf{C})$. Then the family of identities, and the composition operation are specified as

- $id : Obj(\mathbf{C}) \to Mor(\mathbf{C})$, with the constraints that

  $$dom(id_A) = A \text{ and } cod(id_A) = A, \text{ for all } A$$

- $\circ : Comp \to Mor(\mathbf{C})$, where $Comp = \{(g, f) \in Mor(\mathbf{C}) \times Mor(\mathbf{C}) \mid cod(f) = dom(g)\}$, with the constraints that

  $$dom(g \circ f) = dom(f) \text{ and } cod(g \circ f) = cod(g)$$

  and subject to the above monoid equations, in which we assume that all compositions are defined.

From the data of definition 1.1.1, we retrieve $dom(f)$ from the tag of $f.(A, B)$ in the disjoint union $Mor(\mathbf{C}) = \{f.(A, B) \mid A \in Obj(\mathbf{C}), B \in Obj(\mathbf{C}), f \in \mathbf{C}[A, B]\}$. (We recall that the disjoint union of two sets X, Y is defined as, say $X.1 \cup X.2$, where, say, $X.1 = \{x.1 \mid x \in X\}$.) Conversely, given $Mor(\mathbf{C})$, we retrieve $\mathbf{C}[A, B]$ as $\{f \in Mor(\mathbf{C}) \mid dom(f) = A \text{ and } cod(f) = B\}$.

This alternative definition of category stresses categories as "graphs with composition (and identities)".

In this book, we shall not be concerned with problems of size. But as the examples will show, $Obj(\mathbf{C})$ or even $\mathbf{C}[A, B]$ may not be sets. When $\mathbf{C}[A, B]$ is a set, for all A, B, we say that $\mathbf{C}$ is locally small, and when moreover $Obj(\mathbf{C})$ is a set, we say that $\mathbf{C}$ is small.

There are two special cases of categories which are of interest:

- If, for all A, B, $\mathbf{C}[A, B]$ has at most one element, then $\mathbf{C}$ amounts to a preorder on the collection of its objects, where $A \leq B$ if and only if $\mathbf{C}[A, B] \neq \emptyset$. (A preorder is a set equipped with a binary reflexive and transitive relation.)

Indeed, the existence of the identities says that $\leq$ is reflexive. Transitivity is shown as follows: if $A \leq B$ and $B \leq C$ then there exists $f : A \to B$ and $g : B \to C$. Hence $\mathbf{C}[A, C]$ is not empty since it contains $g \circ f$.

If moreover $\mathbf{C}[A, B]$ is non-empty only if $A = B$, then we say that $\mathbf{C}$ is *discrete*: its only morphisms are the identity morphisms.

- if $\mathbf{C}$ has a single object A, then $\mathbf{C}$ amounts to a monoid $\mathbf{C}[A, A]$. This further justifies the intuition of a category as a "typed monoid".

The degenerated preorder case is important, as it will often be easy to see a general categorical definition as a generalisation of a known concept. For example, limits will generalise greatest lower bounds.

Here are some "real" categories:

- **Set**: the objects are sets, the morphisms are functions. This is a locally small category, but not a small category, since the sets form a proper class.
- Algebraic structures: for example, groups and group homomorphisms form a category, ring and ring homomorphisms form a category, etc. . . .
- Topological spaces and continuous functions.
- Preorders and monotonic functions.

Here are some categories where the morphisms are not functions.

- **PSet**. The objects are sets, and the morphisms are partial functions, i.e.

$$\mathbf{PSet}[X, Y] = \{(X', f) \mid X' \subseteq X \text{ and } f \in \mathbf{Set}[X', Y]\}$$

- **Rel**. The objects are sets, the morphisms are binary relations, with $x \; id_X \; y$ if and only if $x = y$ and, for R a relation from X to Y, S a relation from Y to Z:

$$x \; (S \circ R) \; z \quad \text{if and only if there exists} \quad z \in Y \;\text{ such that }\; x \; R \; y \;\text{ and }\; y \; S \; z$$

We now list a few fundamental operations for constructing new categories.

Definition 1.1.2 (dual category) *Let* $\mathbf{C}$ *be a category. We define the dual category* $\mathbf{C}^{op}$ *as follows:*

$$\begin{array}{ll} Obj(\mathbf{C}^{op}) = Obj(\mathbf{C}) & \mathbf{C}^{op}[A, B] = \mathbf{C}[B, A] \\ id^{op} = id & f \circ^{op} g = g \circ f \;\; . \end{array}$$

Given a property P for a category $\mathbf{C}$ and relative theorems, it will often make sense to consider a *dual property* P^{op} to which correspond *dual theorems*. This idea can be formalised using the notion of dual category as follows: given a property P, we say that $\mathbf{C}$ has property P^{op} if $\mathbf{C}^{op}$ has property P.

Definition 1.1.3 (subcategory) *Let* $\mathbf{C}$ *be a category. A subcategory of* $\mathbf{C}$ *is a category* $\mathbf{C}'$ *such that* $Obj(\mathbf{C}') \subseteq Obj(\mathbf{C})$, $\mathbf{C}'[A', B'] \subseteq \mathbf{C}[A', B']$ *for all* $A', B' : \mathbf{C}'$ *and such that the identities and composition operation on* $\mathbf{C}'$ *are the restrictions of those of* $\mathbf{C}$*. If moreover* $\mathbf{C}'[A', B'] = \mathbf{C}[A', B']$ *for all* $A', B' : \mathbf{C}'$*, then* $\mathbf{C}'$ *is called a full subcategory of* $\mathbf{C}$*.*

In terms of graphs, a subcategory is a subgraph closed under composition and identities.

Definition 1.1.4 *If* $\mathbf{C}$ *and* $\mathbf{D}$ *are categories, the* product category $\mathbf{C} \times \mathbf{D}$ *is defined by:*

$$\begin{array}{ll} Obj(\mathbf{C} \times \mathbf{D}) = Obj(\mathbf{C}) \times Obj(\mathbf{D}) & (\mathbf{C} \times \mathbf{D})[(A, B), (A', B')] = \mathbf{C}[A, A'] \times \mathbf{D}[B, B'] \\ id_{(A,B)} = (id_A, id_B) & (f', g') \circ (f, g) = (f' \circ f, g' \circ g) \end{array}$$

The following definition spells some remarkable properties of morphisms.

Definition 1.1.5 *Let* $\mathbf{C}$ *be a category.*

- *A morphism* $f : A \to B$ *is a* monomorphism *(or is a mono, or is mono) if*

$$\forall C, h : C \to A, k : C \to A \;\; (f \circ h = f \circ k \Rightarrow h = k)$$

- *A morphism* $f : A \to B$ *is an* epimorphism *(epi for short) if it is mono in* $\mathbf{C}^{op}$*, i.e.,*

$$\forall C, h : B \to C, k : B \to C \;\; (h \circ f = k \circ f \Rightarrow h = k)$$

- *A morphism* $f : A \to B$ *is a* split mono *if there is a morphism* $g : B \to A$ *such that* $g \circ f = id$*. Dually, a morphism* $g : B \to A$ *is a* split epi *if there is a morphism* $f : A \to B$ *such that* $g \circ f = id$*.*

- *A morphism* $f : A \to B$ *is an* isomorphism *(iso for short) if there is a morphism* $g : B \to A$ *(called the inverse of* f*) such that* $g \circ f = id$ *and* $f \circ g = id$*. We write* $A \cong B$ *if there is an iso between* A *and* B*.*

It should be clear that a split mono is mono, that a split epi is epi, and that a morphism is iso if and only if is mono and split epi (or epi and split mono).

Exercise 1.1.6 *Show that in* **Set** *the monos are the injective functions and the epis are the surjective functions. (See also Exercise 1.3.5).*

Exercise 1.1.7

1. *Show that, in the category of rings and ring morphisms, the inclusion from* $\mathbb{Z}$ *to* $\mathbb{Q}$ *is a non-surjective epi, and is also an example of an epi and mono that is not invertible.*

2. *Idem for the inclusion of* $\mathbb{Q}$ *in* $\mathbb{R}$ *in the category of (separated) topological spaces and continuous functions.*

1.2 Limits and colimits

We now turn to the important notions of limit and colimit. We first give examples, and then a general definition.

Definition 1.2.1 (terminal object) *An object A in a category* $\mathbf{C}$ *is* terminal *if $\forall b \in \mathbf{C}\ \exists! f : B \to A$. We denote a terminal object with 1 and with $!_B$ the unique morphism from B to 1.*

Definition 1.2.2 (product) *Let A, B be two objects in a category* $\mathbf{C}$. *A* product *of A, B is a triple $(C, \pi_1 : C \to A, \pi_2 : C \to B)$ such that for any triple $(D, f : D \to A, g : D \to B)$ there exists a unique $h : D \to C$ such that $\pi_1 \circ h = f$ and $\pi_2 \circ h = g$. We write $C = A \times B$ and $h = \langle f, g \rangle$ (the pair of f, g).*

The situation is illustrated by the following diagram.

$$\begin{tikzcd}
 & D \arrow[d, dashed, "\langle f,g \rangle"'] \arrow[ddl, squiggly, bend right, "f"'] \arrow[ddr, squiggly, bend left, "g"] & \\
 & A \times B \arrow[dl, "\pi_1"] \arrow[dr, "\pi_2"'] & \\
A & & B
\end{tikzcd}$$

where we have used the following conventions: the fat arrows form the structure being defined, the ondulating arrows correspond to the universal quantification ("for all D, f, g") and the dashed arrow to the (uniquely) existing arrow h.

Exercise 1.2.3 *Show that the following purely equational description of the product is equivalent to that given in the Definition 1.2.2:*

$$\begin{aligned}
&\pi_1 \circ \langle f, g \rangle = f \\
&\pi_2 \circ \langle f, g \rangle = g \\
&\langle f \circ \pi_1, f \circ \pi_2 \rangle = f
\end{aligned}$$

Exercise 1.2.4 *Show that the equational presentation of Exercise 1.2.3 is equivalent to the following one (i.e. the two equational theories they define are equal):*

$$\begin{aligned}
&\pi_1 \circ \langle f, g \rangle = f \\
&\pi_2 \circ \langle f, g \rangle = g \\
&\langle f, g \rangle \circ h = \langle f \circ h, g \circ h \rangle \\
&\langle \pi_1, \pi_2 \rangle = id
\end{aligned}$$

Definition 1.2.5 (Equaliser) *Let A, B be two objects, and f, g be two morphisms between A and B, in some category* $\mathbf{C}$. *An* equaliser *of f, g is a pair $(C, e : C \to A)$ such that $f \circ e = g \circ e$, and such that, for all $(C', e' : C' \to A)$, if $f \circ e' = g \circ e'$ then $\exists ! z : C' \to C\ \ (e \circ z = e')$.*

The following picture illustrates equalisers.

$$\begin{tikzcd}
C' \arrow[d, dashed, "z"] \arrow[dr, squiggly, "e'"] & & \\
C \arrow[r, "e"'] & A \arrow[r, bend left, "g"] \arrow[r, bend right, "f"'] & B
\end{tikzcd}$$

Definition 1.2.6 (Pull-back) *Let A, B, D be objects, and let $f : A \to D$, $g : B \to D$, in some category $\mathbf{C}$. A pullback of f, g is a triple $(C, h : C \to A, k : C \to B)$ such that $f \circ h = g \circ k$, and such that, for all $C', h' : C' \to A, k' : C' \to B)$, if $f \circ h' = g \circ k'$ then $\exists ! z : C' \to C \ \ (h \circ z = h'$ and $k \circ z = k')$.*

The following picture illustrates the definition of pullback. The square in the picture is called a pullback square.

$$\begin{tikzcd}
C' \arrow[dr, dashed, "z"] \arrow[drr, squiggly, bend left, "h'"] \arrow[ddr, squiggly, bend right, "k'"'] & & \\
& C \arrow[r, "h"'] \arrow[d, "k"'] & A \arrow[d, "f"'] \\
& B \arrow[r, "g"] & D
\end{tikzcd}$$

Pullback squares can be put side by side and the result is a pullback (let's call it a rectangle!).

Proposition 1.2.7 *Let $\mathbf{C}$ be a category, and consider the following commuting diagram (i.e. $f \circ h = g \circ f'$ and $f' \circ h' = g' \circ f''$):*

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

1. *If the two squares on the picture are pullbacks, then so is the outer rectangle.*

2. *If the right square and the outer rectangle are pullbacks, then the left square is a pullback.*

Proof. We leave (1) to the reader. As for (2), let $C, k : C \to A', l : C \to B''$ be such that $f' \circ k = g' \circ l$. Using that the outer rectangle is a pullback, we get a unique $z : C \to A''$ such that $f'' \circ z$ and $(hh') \circ z = hk$:

$$\begin{tikzcd}
C \arrow[dr, dashed, "z"] \arrow[drr, bend left, "k"] \arrow[drrr, bend left=40, squiggly, "hk"] \arrow[ddr, bend right, squiggly, "l"'] & & & \\
& A'' \arrow[r, "h'"'] \arrow[d, "f''"'] & A' \arrow[r, "h"'] \arrow[d, "f'"'] & A \arrow[d, "f"'] \\
& B'' \arrow[r, "g'"] & B' \arrow[r, "g"] & B
\end{tikzcd}$$

If we show that in fact $h' \circ z = k$, we shall a fortiori have a unique z relative to the left square, and we will be done. This equality follows from the fact that both $h' \circ z$ and k are mediating for hk and $g'l$ (relatively to the right square):

$$\begin{tikzcd}
C \arrow[dr, dashed, "z"] \arrow[drr, bend left, dashed, "k"] \arrow[drrr, bend left=40, squiggly, "hk"] \arrow[ddr, bend right, squiggly, "l"'] & & & \\
& A'' \arrow[r, dashed, "h'"'] \arrow[d, "f''"'] & A' \arrow[r, "h"'] \arrow[d, "f'"'] & A \arrow[d, "f"'] \\
& B'' \arrow[r, squiggly, "g'"] & B' \arrow[r, "g"] & B
\end{tikzcd}$$

□

We are now ready for the general notion of limit.

Definition 1.2.8 (diagram) *Let* $\mathbf{C}$ *be a category and* $\mathbf{I}$ *be a graph. (By graph, we mean here the same as a category, but without identities and composition, thus we just have* $Obj(\mathbf{I})$*, and for every* $i,j{:}\mathbf{I}$*, a set* $\mathbf{I}[i,j]$ *of edges.) A diagram in* $\mathbf{C}$ *over* $\mathbf{I}$ *is a graph morphism* $D : \mathbf{I} \to \mathbf{C}$*, i.e. it maps every* $i : \mathbf{I}$ *to some object* Di *of* $\mathbf{C}$ *and every* $u : i \to j$ *to some morphism* $Du : Di \to Dj$*.*

Note that the identity of the nodes and edges of $\mathbf{I}$ does not matter, since it serves only for indexing matters. For this reason, (the isomorphism class of) $\mathbf{I}$ is called the shape of $D : \mathbf{I} \to \mathbf{C}$. Below, we shall draw some shapes as graphs with unnamed vertices and edges.

Definition 1.2.9 (cone, limit) *Let* $\mathbf{C}$ *be a category and* $D : \mathbf{I} \to \mathbf{C}$ *be a diagram. A* cone *over* D *is given by an object* C *(the vertex of the cone) and a collection of morphisms* $\lambda_i : C \to Di$ *indexed over* $Obj(\mathbf{I})$ *such that the following triangle commutes (for every edge of* $\mathbf{I}$*):*

$$\begin{tikzcd}
& C \arrow[dl, "\lambda_i"'] \arrow[dr, "\lambda_j"] & \\
Di \arrow[rr, "Du"] & & Dj
\end{tikzcd}$$

We say that the morphism λ_i is the component of the cone at i.

A limiting cone *(or limit for short) for D is a cone $(C, \{\lambda_i\}_{i:\mathbf{I}})$ such that for any other cone given by C' and a family of λ'_i's there exists a unique morphism $z : C' \to C$ (called the mediating morphism) such that all triangles*

$$\begin{tikzcd}
C' \arrow[rr, dashed, "z"] \arrow[dr, squiggly, "\lambda'_i"'] & & C \arrow[dl, "\lambda_i"] \\
& Di &
\end{tikzcd}$$

(indexed over $Obj(\mathbf{I})$) commute. By abuse of language, one often says that C is the limit of the Di's.

We say that $\mathbf{C}$ has all limits of shape $\mathbf{I}$ when all diagrams $D : \mathbf{I} \to \mathbf{C}$ have a limit.

We can now recognise the terminal object, products, equalisers and pullbacks as limits of the following respective shapes:

- Terminal. The shape is empty (i.e., $Obj(\mathbf{I}) = \emptyset :$ then a cone is reduced to its vertex C.

- Product.The shape is

  $$\bullet \qquad \bullet$$

  i.e., $\mathbf{I}$ has two vertices and no edge. More generally, the product of any indexed collection of objects (notation $\Pi_{i \in I} A_i$) is defined as the limit of the diagram $D : \mathbf{I} \to \mathbf{C}$ (where $\mathbf{I}$ is the graph that has I as set of vertices, and that has no edges) defined by $Di = A_i$. We stress the case of the product of two objects by calling it a binary product. When I is finite of cardinal n and all A_i's are equal, i.e., $A_i = A$ for some A, we write $\Pi_{i \in I} A = A^n$.

- Equaliser. The shape is

$$\begin{tikzcd} \bullet \arrow[r, bend left] \arrow[r, bend right] & \bullet \end{tikzcd}$$

- Pull-back. The shape is

$$\begin{tikzcd} & \bullet \arrow[d] \\ \bullet \arrow[r] & \bullet \end{tikzcd}$$

We spell out the pullback case. There are three vertices and two edges in the shape, let us name them (different names for different vertices, different names for different edges):

$$\begin{tikzcd} & i \arrow[d, "u"] \\ j \arrow[r, "v"'] & k \end{tikzcd}$$

Then we recover the data of definition 1.2.6 by defining D as follows:

$$Di = A \quad Dj = B \quad Dk = D \quad Du = f \quad Dv = g$$

Note that neither A, B, D nor f, g are imposed to be distinct. And this is precisely why we use this indirect machinery of a separate graph $\mathbf{I}$ rather than just taking a subgraph of $\mathbf{C}$.

A cone over this diagram, strictly speaking, consists of three morphisms $\lambda_i : C \to A$, $\lambda_j : C \to B$, and $\lambda_k : C \to D$ satisfying

$$f \circ \lambda_i \quad = \quad \lambda_k \quad = \quad g \circ \lambda_j \ ,$$

but we see that the middle morphism λ_k is superfluous, since it is defined in terms of either of the two others, and that all information is in the equality $f \circ \lambda_i = g \circ \lambda_j$, as in definition 1.2.6 (with $\lambda_i = h$, $\lambda_j = k$).

(Similarly, for equalisers, the morphism to the common target B is superfluous, and a cone thus reduces to a morphism $e' : C' \to A$.)

Exercise 1.2.10 *Show that a category with terminal object and pullbacks has binary products and equalisers.*

Exercise* 1.2.11 *Show that a category which has all equalisers and all finite products has all finite limits (here finite means that the shape of the diagram has finitely many vertices and edges).*

Exercise 1.2.12 *Show that* **Set** *has all limits and all colimits (hint: limits are subsets of a product, colimits are quotients of a disjoint union).*

The dual notion of limit is that of colimit. Let $D : \mathbf{I} \to \mathbf{C}$ be a diagram. A cocone over D is given by a vertex C and a collection of morphisms $\lambda_i : Di \to C$ (that go thus now *into* C) such that

$$\forall\, u : i \to j \;\; \lambda_j \circ Du = \lambda_i \; .$$

Such a cocone is colimiting if, for every other cocone $(C', \{\lambda'_i\}_{i:\mathbf{I}})$, there exists a unique mediating $z : C \to C'$ (thus *from* C) such that

$$\forall\, i \;\; z \circ \lambda_i = \lambda'_i \; .$$

Since "colimiting cocone" is a bit ugly, one often speaks of cones for cocones.

Here are the duals of the special cases considered above:

- Initial. An initial object 0 is a colimit of the empty diagram. This just means that, for evey object $A : \mathbf{C}$, $\mathbf{C}[0, A]$ consists of exactly one morphism.

- Coproduct. A coproduct is a colimit of a diagram of shape

$$\bullet \qquad \bullet$$

  Let us spell out the definition: a coproduct of A, B consists of a triple

$$(C, i_1 : A \to C, i_2 : B \to C)$$

  such that for any triple $(D, f : A \to D : B \to D)$ there exists a unique $h : C \to D$ such that $h \circ i_1 = f$ and $h \circ i_2 = g$. We write $C = A + B$ and $h = [f, g]$ (the copair of f, g). We use the notation $\coprod_{i \in I} A_i$ for a coproduct of a family of objects.

- Coequaliser. A coequaliser is a colimit of a diagram of shape

$$\begin{tikzcd} \bullet & \bullet \arrow[l, bend right] \arrow[l, bend left] \end{tikzcd}$$

- Push-out. A pushout is a colimit of a diagram of shape

$$\begin{tikzcd} & \bullet \\ \bullet & \bullet \arrow[l] \arrow[u] \end{tikzcd}$$

In general, the dual of a limit of shape $\mathbf{I}$ is a colimit of shape $\mathbf{I}^{op}$ (where $\mathbf{I}^{op}$ is defined by $Obj(\mathbf{I}^{op}) = Obj(\mathbf{I})$ and $\mathbf{I}^{op}[i, j] = \mathbf{I}[j, i]$).

An important property of limits and colimits is that they are unique up to unique isomorphism: if $(C, \{\lambda_i\}_{i:\mathbf{I}})$ and $(C', \{\lambda'_i\}_{i:\mathbf{I}})$ are two limiting cones for the same diagram $D : \mathbf{I} \to \mathbf{C}$, then there exists a unique iso $\iota : C \to C'$ such that $\lambda'_i \circ \iota = \lambda_i$, for all i. We leave the easy proof to the reader.

Exercise 1.2.13 *Show that every coequaliser is epi, and that every equalizer is mono. Show that h, k (referring to the definition of pullbacks) are jointly mono (define this notion). More generally, what can we say of a limit or colimit cone?*

We end this section with the notion of cartesian closed category.

Definition 1.2.14 (CCC) *Let* **C** *be a cateogory that has binary products. An* exponent *of two objects A, B is a pair $(C, ev : C \times A \to B)$ such that for any other pair $(C', f : C' \times A \to B)$ there exists a unique arrow $g : C' \to C$ such that the following triangle commutes:*

$$\begin{tikzcd} C \times A \arrow[d, "ev"'] & C' \times A \arrow[l, dashed, "\langle g \circ \pi_1, \pi_2 \rangle"'] \arrow[dl, squiggly, "f"] \\ B & \end{tikzcd}$$

We write $C = B^A$ (or $C = A \Rightarrow B$) and $g = \Lambda(f)$. We call g the curried form of f, and f the uncurried form of g. A cartesian category is a category which has a terminal object and all binary products (and hence all finite products). A cartesian closed category (CCC for short) is a cartesian category in which all pairs of objects have an exponent.

Exercise 1.2.15 *Show that the following purely equational description of the exponent is equivalent to that given in definition 1.2.14 (in presence of products):*

$$\begin{array}{l} ev \circ \langle \Lambda(f) \circ \pi_1, \pi_2 \rangle = f \\ \Lambda(ev \circ \langle g \circ \pi_1, \pi_2 \rangle) = g \end{array}$$

Exercise 1.2.16 *Show that the equational presentation of Exercise 1.2.15 is equivalent to the following one:*

$$\begin{array}{l} ev \circ \langle \Lambda(f), g \rangle = f \circ \langle id, g \rangle \\ \Lambda(f) \circ g = \Lambda(f \circ \langle g \circ \pi_1, \pi_2 \rangle) \\ \Lambda(ev) = id \end{array}$$

There are very few cartesian closed categories in mathematics. Among the "real" categories listed above, only **Set** and the category of preorders and monotonic functions are CCC's. But they nevertheless are fundamental in theoretical computer science, since they are the categorical counterpart of λ-calculus. (see Chapter 5). Many cartesian closed categories have been built for modelling λ-calculus. Cartesian closed categories play also an important role in category theory itself. Two important examples of CCC's are given in Section 1.3.

A more frequent structure in "mainstream mathematics" is that of a monoidal closed category. It is the same definition, where the notion of product is replaced by a weaker notion of monoidal product.

1.3 Functors, natural transformations

We are interested in the following question: is there a category of categories, or can we define a good notion of morphism between categories?

Definition 1.3.1 (functor) *Let* $\mathbf{C}$, $\mathbf{D}$ *be categories. A functor* $F : \mathbf{C} \to \mathbf{D}$ *is a morphism between the underlying graphs that preserves identity and composition, that is, it consists of a function* $F_{Obj} : Obj(\mathbf{C}) \to Obj(\mathbf{D})$ *and a family of functions* $F_{A,B} : \mathbf{C}[A,B] \to \mathbf{D}[FA,FB]$ *such that the following equations hold universally*

$$Fid = id \qquad F(f \circ g) = Ff \circ Fg$$

Note that we freely write F *without subscripts, and Ff rather than F(f), etc.... A functor* $F : \mathbf{C} \to \mathbf{C}$ *is called an endofunctor. A* contravariant *functor* F *from* $\mathbf{C}$ *to* $\mathbf{D}$ *is a functor* $F : \mathbf{C}^{op} \to \mathbf{D}$*. A functor, say,* $F : \mathbf{C_1}^{op} \times \mathbf{C_2} \to \mathbf{D}$ *is called contravariant in its first argument and covariant in its second argument. A frequent notation for a functor of, say, two arguments, is to write* $F(A,f)$ *for* $F(id_A, f)$.

The full specification of a functor can be given in the deductive style:

$$\frac{A : \mathbf{C}}{FA : \mathbf{D}} \quad \frac{f : A \to B}{Ff : FA \to FB} \quad \frac{id_A : A \to A}{Fid_A = id_{FA} : FA \to FA} \quad \frac{f : B \to C \quad g : A \to B}{Ff \circ Fg = F(f \circ g) : FA \to FC}$$

Categories and functors form a category, which we call **Cat**. The identity functor and the composition of functors are defined obviously:

$$idA = A \quad idf = f \qquad (G \circ F)A = G(FA) \quad (G \circ F)f = G(Ff)$$

We shall freely write GF for $G \circ F$, GFA for $G(FA)$, etc.... Note that $idid$ in this context should read as $id_{\mathbf{C}}(id_A)$, for $A : \mathbf{C}$.

An important example of functor is given in the next definition.

Definition 1.3.2 (hom-functor) *Let* $\mathbf{C}$ *be a locally small category. We define the hom-functor* $Hom_{\mathbf{C}} : \mathbf{C}^{op} \times \mathbf{C} \to \mathbf{Set}$ *as follows:*

$$\begin{array}{l} Hom_{\mathbf{C}}(A,B) = \mathbf{C}[A,B] \\ Hom_{\mathbf{C}}(f,g)(h) = g \circ h \circ f \ \ (f : A' \to A, g : B \to B', h : A \to B) \end{array}$$

Note that the hom-functor is contravariant in its first argument and covariant in its second argument. A frequent notation for the hom-functor is $\mathbf{C}[\_,\_]$, where $\_$ appears as a place holder. A slightly better notation is $\mathbf{C}[\_1,\_2]$, with two distinct place holders (since they are to be replaced by distinct entities). We shall also use the λ-calculus inspired notation $\lambda x,y.\mathbf{C}[x,y]$.

Here is another important functor:

Proposition 1.3.3 *Let* $\mathbf{C}$ *be a category that has all binary products. Then* $\times$ *extends to a functor from* $\mathbf{C} \times \mathbf{C}$ *to* $\mathbf{C}$.

PROOF. Let $f : A \to A'$, $g : B \to B'$. We set $f \times g = \langle f \circ \pi_1, g \circ \pi_2 \rangle$. We show, using Exercise 1.2.4, that this defines a functor:

$$\begin{array}{rcl} id \times id = \langle \pi_1, \pi_2 \rangle & = & id \\ (f' \times g') \circ (f \times g) & = & \langle f' \circ (\pi_1 \circ \langle f \circ \pi_1, g \circ \pi_2 \rangle), g' \circ (\pi_2 \circ \langle f \circ \pi_1, g \circ \pi_2 \rangle) \rangle \\ & = & \langle f' \circ f \circ \pi_1, g' \circ g \circ \pi_2 \rangle \\ & = & (f' \circ f) \times (g' \circ g) \end{array}$$

□

The following definition states two important properties for functors.

Definition 1.3.4 *A functor* $F : \mathbf{C} \to \mathbf{D}$ *is called* full *if*

$$\forall A, B \ \forall h : FA \to FB \ \exists f : A \to B \ \ (Ff = h)$$

i.e. if it is surjective on each homset $\mathbf{C}[A, B]$. *It is called* faithful *if it is injective on each hom-set* $\mathbf{C}[A, B]$.

Exercise 1.3.5 *Let* $\mathbf{C}$ *be a category that has a terminal object* 1. *We say that* $\mathbf{C}$ *has enough points if the functor* $\mathbf{C}[1, \_] : \mathbf{C} \to \mathbf{Set}$ *is faithful. Show that in such a category the following equivalence and implication hold:*

$$\begin{array}{lcl} f \ \mathit{mono} & \Leftrightarrow & \mathbf{C}[1, f] \ \mathit{injective} \\ f \ \mathit{epi} & \Leftarrow & \mathbf{C}[1, f] \ \mathit{surjective} \end{array}$$

(The definition of "enough points" amounts to say that every morphism $f : A \to B$ *is characterised by the underlying function from the points of* A *to the points of* B *(we call* point *a morphism of source* 1*). The exercise thus implies that, in usual categories of sets with structure and structure preserving functions, the monos are the injective morphisms, and the surjective morphisms are always epi (but not conversely, cf. Exercise 1.1.7).) (Hint: the faithfulness assumption is used only in the* $\Leftarrow$ *direction.)*

The next definition relates limits and functors.

Definition 1.3.6 (limit preservation) *Suppose that* $F : \mathbf{C} \to \mathbf{D}$ *is a functor, and* $D : \mathbf{I} \to \mathbf{C}$ *is a diagram which has a limit* $(C, \{\lambda_i\}_{i:\mathbf{I}})$. *We say that* F *preserves the limit of* D *if* $(FC, \{F\lambda_i\}_{i:\mathbf{I}})$ *is a limit of the diagram* $F \circ D : \mathbf{I} \to \mathbf{D}$ *(it is clearly a cone over this diagram).*

Exercise 1.3.7 *Show that the functor* $\mathbf{C}[C, \_] : \mathbf{C} \to \mathbf{Set}$ *preserves all limits.*

Now we ask the following question. Let us fix two categories $\mathbf{C}$ and $\mathbf{D}$: is there a category whose objects are the functors $F : \mathbf{C} \to \mathbf{D}$?

Definition 1.3.8 (natural transformation) *Let $F, G : \mathbf{C} \to \mathbf{D}$ be functors. A natural transformation $\mu : F \to G$ is a family of morphisms $\{\mu_A : FA \to GA\}_{A:\mathbf{C}}$ (called the components of μ) such that, for all $A, B, f : A \to B$, the following square commutes:*

$$\begin{tikzcd}
FA \arrow[r, "\mu_A"] \arrow[d, "Ff"'] & GA \arrow[d, "Gf"] \\
FB \arrow[r, "\mu_B"] & GB
\end{tikzcd}$$

Functors from $\mathbf{C}$ to $\mathbf{D}$ and natural transformations between them indeed form a category, which we denote as $\mathbf{D}^{\mathbf{C}}$ (this is justified by Proposition 1.3.11). The identity and the composition of natural transformations are defined as follows:

$$id_A = id_{FA} \qquad (\nu \circ \mu)_A = \nu_A \circ \mu_A$$

(in the first equality, the first occurrence of id refers to the natural transformation being defined, and the second occurrence refers to an identity morphism in $\mathbf{D}$). The composition just defined is called vertical (see below).

A picture that is often shown for a natural transformation $\mu : F \to G$ is the following one.

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

Then a pictorial representation of the vertical composition of two natural transformations looks as follows:

$$\begin{tikzcd}
\mathbf{C} \arrow[r, bend left=70, "F"{name=U}] \arrow[r, "G"{name=M}] \arrow[r, bend right=50, "H"'{name=L}] & \mathbf{D}
\arrow[Rightarrow, from=U, to=M, "\mu"]
\arrow[Rightarrow, from=M, to=L, "\nu"]
\end{tikzcd}$$

In the next chapter we shall consider another pictorial representation, which we shall use extensively.

Exercise 1.3.9 *Show that a natural transformation μ is iso (in $\mathbf{D}^{\mathbf{C}}$) if and only if all its components are iso (in $\mathbf{D}$).*

Exercise 1.3.10 *Let $\mathbf{C}$, $\mathbf{D}$ be categories, and let $\mathbf{I}$ be a graph. Show that if $\mathbf{D}$ has all limits of shape $\mathbf{I}$ then $\mathbf{D}^{\mathbf{C}}$ has all limits of shape $\mathbf{I}$, and that the limit of a diagram $F : \mathbf{I} \to \mathbf{D}^{\mathbf{C}}$ is given by the formula $(\lim F)C = \lim(\lambda x.FxC)$.*

We now have enough material to give our first "categorical" example of a CCC.

Proposition 1.3.11 *The category* **Cat** *is a CCC.*

Proof. The terminal category **1** is the category with just one object and one morphism (the identity morphism on the unique object). We already defined the product of two categories (Definition 1.1.4). The functors π_1 and π_2, and the pairing of two functors are defined in the obvious way:

$$\begin{array}{lll} \pi_1(C,D)=C & \pi_2(C,D)=D & \langle F,G\rangle(C)=(FC,GC) \\ \pi_1(f,g)=f & \pi_2(f,g)=g & \langle F,G\rangle(f)=(Ff,Gf) \end{array}$$

(for short, we may write $\pi_1 = \lambda(x,y).x$, $\pi_2 = \lambda(x,y).y$, and $\langle F,G\rangle = \lambda x.(Fx,Gx)$). We now check that the category $\mathbf{D}^{\mathbf{C}}$ of functors and natural transformations indeed provides an exponent. We define ev as follows. On objects, $ev(F,A) = FA$. On morphisms, $ev(\mu,f)$ is defined as the diagonal of the naturality square:

$$\begin{tikzcd} FA \arrow[r, "\mu_A"] \arrow[d, "Ff"'] \arrow[dr] & GA \arrow[d, "Gf"] \\ FB \arrow[r, "\mu_B"] & GB \end{tikzcd}$$

i.e. $ev(\mu,f) = Gf \circ \mu_A = \mu_B \circ Ff$.

For $F : \mathbf{C}' \times \mathbf{C} \to \mathbf{D}$, we set $\Lambda(F) = \lambda x'.(\lambda x.F(x',x))$, or, with a mixed but may be more suggestive notation,

$$\Lambda(F) = \lambda x'.F(x',\_)$$

Let us spell this out: $F(C',\_)$ is defined by:

- $F(C',\_)C = F(C',C)$ and $F(C',\_)f = F(C',f)$ (notice the handyness of the abuse of notation $F(C',f)$),
- for $f' : C' \to D'$, $F(f',\_) : F(C',\_) \to F(D',\_)$ is defined by $F(f',\_)_C = F(f',C)$. We check the naturality of $F(f',\_)$. The two paths from $F(C',C)$ to $F(D',D)$ in the naturality square

$$\begin{tikzcd} F(C',C) \arrow[r, "F(f'{,}C)"] \arrow[d, "F(C'{,}f)"'] & F(D',C) \arrow[d, "F(D'{,}f)"] \\ F(C',D) \arrow[r, "F(f'{,}D)"] & F(D',D) \end{tikzcd}$$

  are equal to $F(f',f)$. This "implicit naturality" underlying a functor of two arguments is a consequence of functoriality. We obtain the paths $F(D',f) \circ F(f',C)$ and $F(f',D) \circ F(C',f)$ by writing, respectively:

$$(f',f) = (id_{D'} \circ f', f \circ id_C) \quad \text{and} \quad (f',f) = (f' \circ id_C, id_D \circ f)$$

The verification that all this indeed provides the data for a CCC structure is left to the reader. $\square$

Natural transformations have a richer structure than that provided by the above vertical composition. First we define the (horizontal) composition of a natural transformation with a functor. If $G : \mathbf{B} \to \mathbf{C}$, $F, F' : \mathbf{C} \to \mathbf{C}'$ and $\mu : F \to F'$, then we define μG by set theoretical composition, i.e., $(\mu G)_A = \mu_{GA}$ for all A. It is immediate that μG is a natural transformation from FG tp $F'G$. Likewise, if $H : \mathbf{C}' \to \mathbf{B}'$, $F, F' : \mathbf{C} \to \mathbf{C}'$ and $\nu : F \to F'$, then $H\nu : HF \to HF'$ is natural, where $(H\nu)_A = H(\nu_A)$ for all A.

Then, if $F, F' : \mathbf{C} \to \mathbf{C}'$, $G, G' : \mathbf{C}' \to \mathbf{C}''$, $\mu : F \to F'$, and $\nu : G \to G'$ are given, we can make sense of, or parse, the picture

$$\begin{tikzcd} \mathbf{C} \arrow[r, bend left, "F"] \arrow[r, bend right, "F'"'] \arrow[r, phantom, "\Downarrow\mu"] & \mathbf{C}' \arrow[r, bend left, "G"] \arrow[r, bend right, "G'"'] \arrow[r, phantom, "\Downarrow\nu"] & \mathbf{C}'' \end{tikzcd}$$

either as $(\nu F') \circ (G\mu)$ or as $(G'\mu) \circ (\nu F)$, as illustrated by the following pictures:

$$\begin{tikzcd} \mathbf{C} \arrow[r, bend left, "F"] \arrow[r, bend right, "F'"'] \arrow[r, phantom, "\Downarrow\mu"] & \mathbf{C}' \arrow[r, bend left, "G"] \arrow[r, bend right, "G'"'] \arrow[r, phantom, "\Downarrow\nu"] & \mathbf{C}'' \end{tikzcd} \qquad \begin{tikzcd} \mathbf{C} \arrow[r, bend left, "F"] \arrow[r, bend right, "F'"'] \arrow[r, phantom, "\Downarrow\mu"] & \mathbf{C}' \arrow[r, bend left, "G"] \arrow[r, bend right, "G'"'] \arrow[r, phantom, "\Downarrow\nu"] & \mathbf{C}'' \end{tikzcd}$$

where, say, the first picture should read as the vertical composition of

$$\begin{tikzcd} \mathbf{C} \arrow[r, bend left, "F"] \arrow[r, bend right, "F'"'] \arrow[r, phantom, "\Downarrow\mu"] & \mathbf{C}' \arrow[r, bend left, "G"] & \mathbf{C}'' \end{tikzcd}$$

and

$$\begin{tikzcd} \mathbf{C} \arrow[r, bend right, "F'"'] & \mathbf{C}' \arrow[r, bend left, "G"] \arrow[r, bend right, "G'"'] \arrow[r, phantom, "\Downarrow\nu"] & \mathbf{C}'' \end{tikzcd}$$

The two parsings coincide, by naturality of ν (this property is known as Godement's rule):

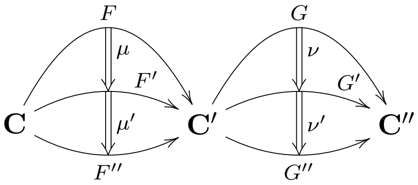
$$\begin{tikzcd} GFA \arrow[r, "\nu_{FA}"] \arrow[d, "G\mu_A"'] & G'FA \arrow[d, "G'\mu_A"] \\ GF'A \arrow[r, "\nu_{F'A}"] & G'F'A \end{tikzcd}$$

This leads us to the following definition:

Definition 1.3.12 *$F, F' : \mathbf{C} \to \mathbf{C}'$, $G, G' : \mathbf{C}' \to \mathbf{C}''$, $\mu : F \to F'$, and $\nu : G \to G'$. We define the horizontal composition $\nu \cdot \mu : GF \to G'F'$ of μ and ν as follows:*

$$(\nu F') \circ (G\mu) \quad = \quad \nu \cdot \mu \quad = \quad (G'\mu) \circ (\nu F)$$

Now consider the following diagram:

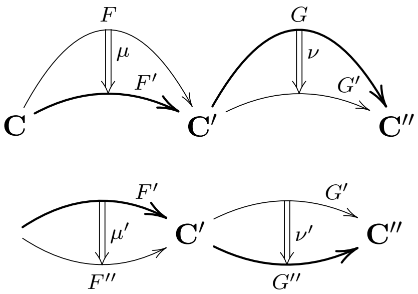


How should we parse it? Should we first make the horizontal compositions, or should we first make the vertical ones? The answer is that it does not matter, and hence that the pictorial representation is acurate, since it abstracts from these details.

Proposition 1.3.13 *Let $F, F', F'' : \mathbf{C} \to \mathbf{C}'$, $G, G', G'' : \mathbf{C}' \to \mathbf{C}''$, $\mu : F \to F'$, $\mu' : F' \to F''$, $\nu : G \to G'$ and $\nu' : G' \to G''$. Then the following equality, called* interchange law*, holds:*

$$(\nu' \circ \nu) \cdot (\mu' \circ \mu) \quad = \quad (\nu' \cdot \mu') \circ (\nu \cdot \mu)$$

Proof. We start from the right hand side. We parse $\nu \cdot \mu$ along the line GF' and we parse $\nu' \cdot \mu'$ along the line $G''F'$:



This give us

$$\begin{aligned}
(\nu' \cdot \mu') \circ (\nu \cdot \mu) &= G''\mu' \circ (\nu' F' \circ \nu F') \circ G\mu \\
&= G''\mu' \circ ((\nu' \circ \nu)F' \circ G\mu) \\
&= (G''\mu' \circ G''\mu) \circ (\nu' \circ \nu)F \quad \text{by Godement's rule} \\
&= G''(\mu' \circ \mu) \circ (\nu' \circ \nu)F \\
&= (\nu' \circ \nu) \cdot (\mu' \circ \mu)
\end{aligned}$$

where we have also used, say, $\nu' F' \circ \nu F' = (\nu' \circ \nu)F'$, which is straightforward to check. $\square$

1.4 Yoneda lemma and embedding

In this section, we study a remarkable class of functors. Let $\mathbf{C}$ be a category. The functors $F : \mathbf{C}^{op} \to \mathbf{Set}$ are called set-valued functors, or *presheaves*. The hom functors $\mathbf{C}[\_, C]$ are called *representable presheaves*.

Lemma 1.4.1 (Yoneda lemma) *For any functor $F : \mathbf{C}^{op} \to \mathbf{Set}$ and any object $C : \mathbf{C}$, the following isomorphism holds in $\mathbf{Set}$:*

$$FC \cong \mathbf{Set}^{\mathbf{C}^{op}}[\mathbf{C}[\_, C], F] \ .$$

PROOF. We define $i_C : FC \to \mathbf{Set}^{\mathbf{C}^{op}}[\mathbf{C}[\_, C], F]$ with inverse $j_C : \mathbf{Set}^{\mathbf{C}^{op}}[\mathbf{C}[\_, C], F] \to FC$ as follows:

$$i_C(x) = \lambda D.\lambda l : D \to C.(Fl)(x) \qquad j_C(\tau) = \tau_C(id_C) \ .$$

We leave it to the reader to prove that $i_C(x)$ is natural and that i_C and j_C are inverse. We refer to section 2.6 for a graphical proof. $\square$

Theorem 1.4.2 (Yoneda) *For any category $\mathbf{C}$, the curried form of the hom functor $Y = \lambda x.\mathbf{C}[\_, x] : \mathbf{C} \to \mathbf{Set}^{\mathbf{C}^{op}}$ is full and faithful. It is called the Yoneda embedding.*

PROOF. We specialise the Yoneda lemma to $F = YC'$, and we verify that $if = Yf$, for all f. Indeed, we have, for all $D, l : D \to C$:

- $(if)_D l = YC'lf = \mathbf{C}[l, C']f = l \circ f$ on one hand,
- $(Yf)_D l = \mathbf{C}[f, D]l = l \circ f$ on the other hand.

Then we know that Y is bijective on $\mathbf{C}[C, C']$, by the Yoneda lemma. $\square$

Proposition 1.4.3 *Every presheaf is a colimit of representable presheaves.*

PROOF. Let $P : \mathbf{C}^{op} \to \mathbf{Set}$. We are looking for a diagram $D : \mathbf{I} \to \mathbf{C}$ such that P will be the limit of $Y \circ D$. We take as $\mathbf{I}$ the graph underlying the category $Elt(P)$ that is defined as follows: objects are pairs (C, p), with $C : \mathbf{C}$ and $p \in PC$. and we set:

$$Elt(P)[(C, p), (C', p')] = \{u \in \mathbf{C}[C, C'] \mid Pup' = p\}$$

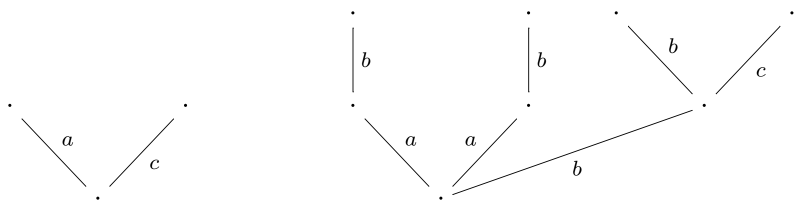
We take $D = \pi$, where $\pi(C, p) = C$ and $\pi(u) = u$.

We construct a cocone $(P, \{h_{(C,p)}\}_{(C,p):Elt(P)}$ over $Y \circ \pi$ by taking $h_{(C,p)} = i_C(p)$, where i_C is given by the Yoneda lemma. Let $(Q, \{\lambda_{(C,p)}\}_{(C,p):Elt(P)}$ be an arbitrary cocone over $Y \circ \pi$ We define $\nu : P \to Q$ by $\nu_C(p) = j(\lambda_{(C,p)})$. We leave it to the reader to show that the colimiting cone is indeed a cocone, and that ν is indeed the unique mediating morphism. We refer to Section 2.6 for a graphical proof. $\square$

The following exercise gives a suggestive illustration of this theorem (in the degenerated case where $\mathbf{C}$ is a preorder).

Exercise 1.4.4 *Let A be an alphabet of letters or* actions*, and let A^* be the set of words over A, ordered by the prefix ordering, and considered as a category.*

1. *Show that giving a functor from $(A^*)^{op}$ to* **Set** *amounts to giving a* synchronisation *forest, i.e. a labelled oriented acyclic graph (in the ordinary sense where there is at most one edge between two vertices). The labels are on edges (and may be non distinct). Here is an example:*



   These forests can be formalised as follows:

   - *A synchronisation forest is a set of synchronisation trees.*
   - *If $\{P_i\}_{i\in I}$ is a family of synchronisation trees and if $\{a_i\}_{i\in I}$ is a family of actions over the same index set, then $\Sigma_{I\in I}\ (a_i \cdot P_i)$ is a synchronisation tree (where the formal sum is taken modulo commutativity and associativity).*

   The base case of the above inductive definition is when $I = \emptyset$. (Synchronisation forests are automata with possibly infinitely many states, without initial and final state, and without loop.) (Hint: think of what should be the action of the functor F on the unique arrow from, say, s to ss' in A^*.)

2. *Spell out what this means that the above example of a synchronisation forest is a colimit of representable functors. (Hint: a forest is a glueing of branches, and glueing has to do with quotienting, which has to do with taking a colimit (cf. Exercise 1.2.12).)*

A key interest of the Yoneda embedding is that it embeds an arbitrary category into a richly structured one. By Exercises 1.2.12 and 1.3.10, $\mathbf{Set}^{\mathbf{C}^{op}}$ has all limits and all colimits. It is also a CCC, as we now show.

Proposition 1.4.5 *The category $\mathbf{Set}^{\mathbf{C}^{op}}$ (for any category $\mathbf{C}$) is a CCC.*

Proof. The cartesian structure is built *pointwise*. The terminal functor maps every object $C : \mathbf{C}$ to the singleton set $\{*\}$, and every morphism to $id_{\{*\}}$. The product $F \times G$ of two functors is defined by $(F \times G)A = FA \times GA$ and $(F \times G)f = Ff \times Gf$. But this would not work for the exponent, because then f would have to behave both contravariantly and covariantly. Yoneda lemma provides the solution. It tells us that $G^F C$ must be in bijective correspondence with $\mathbf{Set}^{\mathbf{C}^{op}}[\mathbf{C}[\_, C], G^F]$, but the latter must be in bijective correspondence with $\mathbf{Set}^{\mathbf{C}^{op}}[\mathbf{C}[\_, C] \times F, G]$. This gives us the definition of G^F on objects:

$$(F \to G)C = \mathbf{Set}^{\mathbf{C}^{op}}[\mathbf{C}[\_, C] \times F, G]$$

The rest follows easily (things come in place by matching the types): For $F, G : \mathbf{C}^{op} \to \mathbf{Set}$ we define:

$$\begin{array}{ll}
((F \to G)f\mu)_A(g, x) = \mu_A(f \circ g, x) & (\mu : \mathbf{C}[_, C] \times F \to G, f : D \to C, \\
 & g : A \to D, x \in Fa) \\
 & \\
ev_C(\mu, x) = \mu_C(id_C, x) & ((\mu : \mathbf{C}[_, C] \times F \to G, x \in FC) \\
 & \\
(\Lambda(\nu)_C z)_A(f, x) = \nu_A(Hfz, x) & (\nu : H \times F \to G, z \in HC, \\
 & f : A \to C, x \in Fa)
\end{array}$$

□

Exercise 1.4.6 *Let* $\mathbf{C}$ *be a CCC which has an initial object* 0*. Then show that for any* $A : \mathbf{C}$*:*

1. $0 \times A \cong 0$,
2. $\mathbf{C}[A, 0] \neq \emptyset$ *implies* $A \cong 0$ *(thus* $\mathbf{C}[A, 0]$ *has at most one element).*
3. *If furthermore* $\mathbf{C}$ *has finite limits, show that, for any* $A : \mathbf{C}$*, the unique morphism from* 0 *to* A *is mono.*

Hints: $\mathbf{C}[0 \times A, B] \cong \mathbf{C}[0, B^A]$ *and consider in particular* $B = 0 \times A$*. Consider also* $!^{op} \circ \pi_1$*. Suppose* $f : A \to 0$*. Then consider* $\pi_2 \circ \langle f, id \rangle$*.*

Exercise 1.4.7 *Let* $\mathbf{C}$ *be a CCC, and* 0 *be an object such that the natural transformation* $\mu : \lambda x.x \to \lambda x.(x \to 0) \to 0$ *defined by* $\mu = \Lambda(ev \circ \langle \pi_2, \pi_1 \rangle)$ *is iso. Show that* 0 *is initial and that* $\mathbf{C}$ *is a preorder*

(This is an important negative fact: there is no nontrivial categorical semantics of classical logic, thinking of 0 *as absurdity and of* $(x \to 0) \to 0$ *as double negation.)*

Hints: (i) $0 \to 0 \cong 1$*, by observing* $1 \cong (1 \to 0) \to 0$*, and* $(1 \to A) \cong A$*, for any* A*. (ii) For any* A*:*

$$\begin{array}{lll}
\mathbf{C}[0, A] & \cong \mathbf{C}[0, (A \to 0) \to 0] & \cong \mathbf{C}[0 \times (A \to 0), 0] \\
 & \cong \mathbf{C}[A \to 0, 0 \to 0] & \cong \mathbf{C}[A \to 0, 1] \ .
\end{array}$$

(iii) For any A, B*:* $\mathbf{C}[A, B] \cong \mathbf{C}[A, (B \to 0) \to 0] \cong \mathbf{C}[A \times (B \to 0), 0]$*.*

1.5 Adjunctions

We now define the notion of adjunction, which has several characterizations.

Definition 1.5.1 (adjunction (definition 1)) *An* adjunction *between two categories* $\mathbf{C}, \mathbf{C}'$ *is a triple* (F, G, ζ), *where* $F : \mathbf{C} \to \mathbf{C}'$ *and* $G : \mathbf{C}' \to \mathbf{C}$ *are functors and* $\zeta : \mathbf{C}'[F_{-1}, {}_{-2}] \to \mathbf{C}[{}_{-1}, G_{-2}]$ *is a natural isomorphism. We say that* F *is the* left adjoint *and that* G *is the* right adjoint, *and we denote this situation by* $F \dashv G$.

Spelling out the definition, we have a family of bijections $\zeta_{C,C'} : \mathbf{C}'[FC, C'] \to \mathbf{C}[C, GC']$ such that

$$\begin{array}{ll} \zeta_{C,D'}(g \circ f) = Gg \circ \zeta_{C,D}(f) & (f : FC \to C', g : C' \to D') \\ \zeta_{D,C'}(f \circ Lh) = \zeta_{C,C'}(f) \circ h & (f : FC \to C, h : D \to C) \end{array}$$

When $\mathbf{C}, \mathbf{C}'$ are preorders, the definition of adjunction boils down to that of a *Galois connection*: two functions F, G such that for all C, C' $FC \leq C'$ if and only if $C \leq GC'$.

Definition 1.5.2 (adjunction (definition 2)) *An* adjunction *between two categories* $\mathbf{C}, \mathbf{C}'$ *is a quadruple* (F, G, η, ϵ), *where* $F : \mathbf{C} \to \mathbf{C}'$ *and* $G : \mathbf{C}' \to \mathbf{C}$ *are functors and* $\eta : id_{\mathbf{C}} \to GF$ *and* $\epsilon : FG \to id_{\mathbf{C}'}$ *are natural transformations such that*

$$G\epsilon \circ \eta G = id_G \quad \textit{and} \quad \epsilon F \circ F\eta = id_L$$

We show how to mutually derive the data of these two definitions:

- Given ζ, we define $\eta_A = \zeta_{A,FA}(id_{FA})$ and $\epsilon_{A'} = \zeta^{-1}_{GA',A'}(id_{GA'})$.
- Given η, ϵ, we define $\zeta_{C,C'}(f) = Gf \circ \eta_C$ and $\zeta^{-1}_{C,C'}(g) = \epsilon_{C'} \circ Fg$.

Definition 1.5.3 (universal morphism) *Let* $G : \mathbf{C} \to \mathbf{D}$ *be a functor and* D *an object in* $\mathbf{D}$. *Then the pair* $(C, u : D \to GC)$ *is* universal *from* D *to* G *if:*

$$\forall C' \ \forall g : D \to GC' \ \exists! f : C \to C' \ (Gf \circ u = g) \ .$$

The notion of a couniversal pair is defined dually.

We note that, in reference to the above definition, if $\iota : D' \to D$ is a an iso, and if $(C, u : D \to GC)$ is universal from D to G, then $(C, u \circ \iota)$ is universal from D' to G: given $g' : D' \to GC'$, we first build $g = g' \circ \iota^{-1}$, and then the unique associated f, which fits and is the only one to fit since $Gf \circ u = g$ reads as $Gf \circ (u \circ \iota) = g'$.

Our last two definitions of adjunction are more economical in the sense that they do not suppose that we have two functors, nor that the unit or counit or the bijections are natural: these come as derived facts.

Definition 1.5.4 (adjunction (definition 3)) *Let* $\mathbf{C}$, $\mathbf{C}'$ *be categories, and let* $G : \mathbf{C}' \to \mathbf{C}$. *Then we say that* G *has a left adjoint if for avery object* $C : \mathbf{C}$ *there exists a universal pair, denoted as* $(FC, \eta_C : C \to GFC)$ *from* C *to* G.

The following diagram (where we use the same conventions as in section 1.2) spells out the definition:

$$\begin{tikzcd} C \arrow[rr, "\eta_C"'] \arrow[dr, squiggly, "g"'] & & GFC \arrow[dl, dashed, "Gf"'] \\ & GC' & \end{tikzcd}$$

We leave it to the reader to extend the definition of F to morphisms, to show that η is then natural (not too surprising, since the definition of F on morphisms is taylored for this), and to check that the correspondence associating f to g defines a bijection between $\mathbf{C}[C, GC']$ and $\mathbf{C}'[FC, C']$, and that this collection of bijections is natural (back to definition 1, with $f = \zeta^{-1}(g)$!)

Definition 3 is useful when we want to construct a left adjoint. One of the most frequent situations of adjunction is the construction of a free structure. We describe this in a very simple instance.

Consider a signature, say, $\Sigma = \{c, f\}$, where c and f are function symbols of respective arities 0 et 2. A Σ-algebra consists of a set A, an element c_A of A for interpreting c and a function f_A from $A \times A \to A$ for interpreting f. We can put these informations together in the form of a morphism $a : (1 + (A \times A)) \to A$. Let $\Sigma : \mathbf{Set} \to \mathbf{Set}$ be the functor defined by $\sigma = \lambda x.\ 1 + (x \times x)$, and more generally $\Sigma = \lambda x. \coprod_{f \in \Sigma} x^{ar(f)}$ for an arbitrary signature, where $ar(f)$ is the arrity of the symbol f. This makes sense in any category that has finite products and coproducts, and in particular in $\mathbf{Set}$. A Σ-algebra is thus a pair $(A, a : \Sigma A \to A)$. A morphism of Σ-algebras is a function between their underlying sets that preserves the (interpretations) of the operations. We leave it to the reader to check that this is synthetically taken care of by the following abstract definition. A morphism from (A, a) to (B, b) is a function $f : A \to B$ such that the following square commutes:

$$\begin{tikzcd} A \arrow[d, "f"] & \Sigma A \arrow[l, "a"'] \arrow[d, "\Sigma f"] \\ B & \Sigma B \arrow[l, "b"] \end{tikzcd}$$

Clearly, Σ-algebras and Σ-algebra morphisms form a category, let us call it $\mathbf{Set}^\Sigma$. More generally, given *any* category $\mathbf{C}$ and any endofunctor $F : \mathbf{C} \to \mathbf{C}$, we can define the category $\mathbf{C}^\Sigma$ of Σ-algebras (see Section 4.1). There is an obvious functor $U : \mathbf{Set}^\Sigma \to \mathbf{Set}$ defined by $U(A, a) = A$ and $Uf = f$. This functor forgets the Σ-algebra structure, and is therefore called a *forgetful* functor. In the reverse direction, there is a functor T_Σ that maps a set X to the *free* Σ-algebra over X, which has as carrier the set of *terms* written with the following syntax:

$$t ::= x \mid c \mid f(t,t)$$

where x ranges over X. The distinguished element of T_Σ is the term c, and the binary operation associated with f maps (t_1,t_2) to $f(t_1,t_2)$. We have $T_\Sigma \dashv U$ and the adjunction says precisely that T_Σ is a free construction:

$$\begin{tikzcd} X \arrow[rr, "\eta_X"] \arrow[dr, squiggly, "g"'] & & UT_\Sigma(X) \arrow[dl, dashed, "Ug^*"] \\ & U(A,a) & \end{tikzcd}$$

Here η_X is the inclusion function: each $x \in X$ is a term, and the unique g^* is the unique extension of g to all terms, which is defined by induction:

$$\frac{}{g^*(x) = g(x)} \qquad \frac{}{g^*(c) = c_A} \qquad \frac{g^*(t_1) = y_1 \quad g^*(t_2) = y_2}{g^*(f(t_1,t_2)) = f_A(y_1,y_2)}$$

Note that in common mathematical practive, the functor U is omitted, and one draws simply:

$$\begin{tikzcd} X \arrow[rr, "\eta_X"] \arrow[dr, squiggly, "g"'] & & T_\Sigma(X) \arrow[dl, dashed, "g^*"] \\ & (A,a) & \end{tikzcd}$$

which is easier to read, while the categorical formulation is more careful about type coercions: an element $x \in X$ is "coerced" to a term $x \in T_\Sigma(X)$, etc....

Our last definition is tailored to the construction of right adjoints.

Definition 1.5.5 (adjunction (definition 4)) *Let* $\mathbf{C}$, $\mathbf{C}'$ *be categories, and let* $F : \mathbf{C} \to \mathbf{C}'$. *Then we say that* F *has a right adjoint if for every object* $C' : \mathbf{C}'$ *there exists a couniversal pair, denoted as* $(GC', \epsilon_{C'} : FGC' \to C')$ *from* F *to* C'.

The following diagram spells out the definition (of course, here $g = \zeta(f)$ in the terminology of the first definition):

$$\begin{tikzcd} FGC' \arrow[dr, "\epsilon_{C'}"'] & & FC \arrow[ll, dashed, "Fg"'] \arrow[dl, squiggly, "f"] \\ & C' & \end{tikzcd}$$

We did not display the diagrams corresponding to the third and fourth definitions symmetrically, but they actually are, up to a rotation. We have drawn the triangles following rather common practice found in papers and books. The fact is that in a given application, one rarely sees the two sorts of diagrams together, so there is no real need to symmetrise them perfectly.

Here are two fundamental examples of adjunctions, for which the fourth presentaiion is the most natural.

- Limits. We fix a graph $\mathbf{I}$ and a category $\mathbf{C}$. A diagram $D : \mathbf{I} \to \mathbf{C}$ is like a functor, less the preservation of identities and composition. The notion of natural transformation does not involve any category structure on the source category, but only on the target category. Therefore it makes perfect sense to speak about natural transformations between diagrams. Diagrams and natural transformations between them form a category, which we denote with $\mathbf{C}^{\mathbf{I}}$. A cone $(C, \{\lambda_i\}_{i:\mathbf{I}})$ over D can be seen as a natural transformation $\lambda : \Delta C \to D$, where $\Delta : \mathbf{C} \to \mathbf{C}^{\mathbf{I}}$ is the curried form of the first projection, i.e., maps C to the constant diagram that maps every i to C and every u to id_C. Then the fact that $\mathbf{C}$ admits all $\mathbf{I}$-limits can be characterised as the fact that the functor Δ has a right adjoint Lim. $LimD$ provides us with the vertex of the limiting cone, and the counit provides us with the limiting cone. We leave the reader convince himself by some drawings. If one is just interested in the limit of some diagram D, then a limiting cone is a couniversal arrow from Δ to D.

  Similarly, colimits give rise to a left adjoint to the constant functor, or individually are universal arrows from D to Δ.

- Exponents. A category has all exponents of the form B^A, for some fixed A, if and only if the functor $\_ \times A$ has a right adjoint G. The morphisms ev form the counit of this adjunction, GB is B^A, and $\Lambda(f)$ is the unique $g : C \to B^A$ associated with $f : C \times A \to B$.

We summarise below the three examples of adjunctions given in this section:

free Σ-algebras	Limits	CCC
$T_\Sigma \dashv U$	$\Delta \dashv Lim$	$(\_ \times A) \dashv \_^A$

We end the section with an important result that relates adjunctions with limit preservation.

Proposition 1.5.6 *Right adjoints preserve limits, and left adjoints preserve colimits.*

Proof. We deal only with limits (the other half of the statement is dual, and hence comes for free). Let $D' : \mathbf{I} \to \mathbf{C}'$ of limit C'. The key is to "translate" a cone $(C, \{\lambda_i : C \to GD'i\}_{i:\mathbf{I}}$ into a cone of vertex FC over D', and to translate back its mediating arrow from FC to C' into an arrow from C to GC'. We leave the details to the reader. (See Section 2.3 for a graphical proof.)

□

We cannot resist giving an alternative, more diagrammatic proof of this result, *under the assumption that both* $\mathbf{C}$ *and* $\mathbf{C}'$ *have all limits of shape* $\mathbf{I}$. This additional assumption allows us to exploit the characterisation of limits by adjunctions. Contemplate the following square of adjunctions (for the bottom adjunction, see Exercise 1.5.9):

$$\begin{tikzcd}
\mathbf{C} \arrow[r, bend left, "G"] \arrow[r, bend right, "F"'] \arrow[d, bend right, "\Delta"'] \arrow[d, bend left, "Lim"] & \mathbf{C}' \arrow[d, bend right, "\Delta"'] \arrow[d, bend left, "Lim"] \\
\mathbf{C}^{\mathbf{I}} \arrow[r, bend left, "G^{\mathbf{I}}"] \arrow[r, bend right, "F^{\mathbf{I}}"'] & \mathbf{C}'^{\mathbf{I}}
\end{tikzcd}$$

We have $\Delta F = F^{\mathbf{I}}\Delta$, as highlighted in the drawing. Indeed, we have

$$\begin{array}{l}
(\Delta F)Ci = \Delta(FC)i = FC \\
(F^{\mathbf{I}}\Delta)Ci = F^{\mathbf{I}}(\Delta C)i = (F \circ (\Delta C))i = F(\Delta Ci) = FC
\end{array}$$

From this it follows (using Exercise 1.5.8) that $GLim$ and $LimG^{\mathbf{I}}$ are both right adjoint to the same functor. Hence they coincide up to isomorphism (see Exercise 1.5.7). Spelling this out, we have proved our statement:

$$G(LimD) = Lim(G \circ D)$$

Exercise 1.5.7 *Show that it* $F \dashv G$ *and* $F \dashv G'$*, then* G *and* G' *coincide up to a unique isomorphism respecting some property (which one?). (We give a graphical solution to this exercise in Section 2.2.)*

Exercise 1.5.8 *Show that adjunctions compose: if* $F \dashv G$ *(with* $F : \mathbf{C} \to \mathbf{C}'$*) and* $F' \dashv G'$ *(with* $F' : \mathbf{C}' \to \mathbf{C}''$*), then* $(F' \circ F) \dashv (G' \circ G)$*.*

Exercise 1.5.9 *Show that if* $F \dashv G$ *(with* $F : \mathbf{C} \to \mathbf{C}'$*), and if* $\mathbf{I}$ *is a graph, then* $F^{\mathbf{I}} \dashv G^{\mathbf{I}}$*, where, say,* $F^{\mathbf{I}} : \mathbf{C}^{\mathbf{I}} \to \mathbf{C}'^{\mathbf{I}}$ *is defined by* $F^{\mathbf{I}}(D) = F \circ D$*. (We give a graphical solution to this exercise in Section 2.3.)*

1.6 Equivalences of categories

In this section, we investigate special cases of adjunctions: equivalences, reflections and coreflections. All these notions have to do with one or two of the functors in a pair of adjoint functors being full and faithuful. We therefore begin by a characterisation of the faithfulness and the fullness of adjoint functors. The proofs of the following two propositions are given in Section 2.4.

Proposition 1.6.1 *For an adjunction $F \dashv G$ the following holds:*

1. *G is faithful if and only if every component of the counit is an epi.*
2. *G is full if and only if every component of the counit is a split mono.*
3. *G is full and faithful if and only if the counit is iso.*

Dually, F is faithful (resp. full) if every component of the unit is mono (resp. split epi), and is full and faithful iff η is iso.

Proposition 1.6.2 *The following properties of a functor $F : \mathbf{C} \to \mathbf{C}'$ are equivalent:*

1. *There exists a functor $G : \mathbf{C}' \to \mathbf{C}$ and two natural equivalences $\iota : GF \to id_{\mathbf{C}}$ and $\iota' : FG \to id_{\mathbf{C}'}$.*
2. *F is part of an adjunction $F \dashv G$ in which the unit and the counit are natural isomorphisms.*
3. *F is full and faithful and $\forall\, C' : \mathbf{C}'\ \exists C \in \mathbf{C}\ \ (C' \cong FC)$.*

When either of these properties holds, we say that F or that $F \dashv G$ is an equivalence of categories.

We note that if $F \dashv G$ is an equivalence of categories, then we also have $G \dashv F$, with unit (resp. counit) the inverse of the counit (resp. of the unit) of the adjunction $F \dashv G$.

Exercise 1.6.3 *Give examples of equivalent but not isomorphic preorders.*

Exercise 1.6.4 *Show that any adjunction cuts down to an equivalence between the full subcategory whose objects are those at which the counit and the unit, respectively, are iso.*

Definition 1.6.5 *A reflection is an adjunction $F \dashv G$ where F is full and faithful (or equivalently where the unit is iso). Dually, a coreflection is an adjunction $F \dashv G$ where G is full and faithful (or equivalently where the counit is iso).*

(In some books, starting with [36], one only assumes the faithfulness of F (resp. G) in the definition of a reflection (resp. coreflection). We do not see much interest in this relaxation, though, as we regard the following proposition as the key property of reflections and coreflections.)

Proposition 1.6.6 *Let $F : \mathbf{C} \to \mathbf{C}'$ and $G : \mathbf{C}' \to \mathbf{C}$ be two functors.*

1. *If $F \dashv G$ is a reflection, if $D' : \mathbf{I} \to \mathbf{C}'$ is a diagram in $\mathbf{C}'$, and if GD' has a colimit in $\mathbf{C}$, then $F(\mathrm{colim}\ GD')$ is a colimit of D'.*

2. *If $F \dashv G$ is a coreflection, if $D : \mathbf{I} \to \mathbf{C}$ is a diagram in $\mathbf{C}$, and if FD has a limit in $\mathbf{C}'$, then $G(\lim FD)$ is a limit of D.*

3. *If $F \dashv G$ is an equivalence of categories, then $\mathbf{C}$ and $\mathbf{C}'$ are interchangeable as regards having limits and colimits.*

Proof. The proofs of (1) and (2) are dual (since an adjunction $F \dashv G$ gives rise to an adjunction $G^{op} \dashv F^{op}$). Statement (3) is a consequence of the observation that when $F \dashv G$ is an equivalence of categories, then $F \dashv G$ and $G \dashv F$ are both a reflection and a coreflection. We prove, say, (2). Since right adjoints preserve limits (Proposition 1.5.6), $G(\lim FD)$ is a limit of GFD. But D and GFD are isomorphic diagrams, since ϵD is iso. Hence $G(\lim FD)$ is a limit of D. $\square$

1.7 Monads

If we have an adjunction $F \dashv G$ between two categories $\mathbf{C}$ and $\mathbf{C}'$, what can we say that concerns only $\mathbf{C}$? First, there is an endofunctor $T = GF : \mathbf{C} \to \mathbf{C}$. Second, the unity reads now as $\eta : id \to T$. Finally, by pre and postcomposing ϵ, we obtain a natural transformation $\mu = G\epsilon F : TT \to T$. This leads us to the following definition.

Definition 1.7.1 *Let $\mathbf{C}$ be a category. A* monad *on $\mathbf{C}$ is a triple (T, η, μ) where $T : \mathbf{C} \to \mathbf{C}$, $\eta : id_{\mathbf{C}} \to T$ and $\mu : TT \to T$, and where η and μ satisfy the following three equations:*

$$\mu \circ (\mu T) = \mu \circ (T\mu) \qquad \mu \circ (\eta T) = id_T \qquad \mu \circ (T\eta) = id_T$$

We say that a functor T gives rise to a monad structure if there exists η, μ that turn it into a monad.

The notion of monad is closely related to that of adjunction. This will be explained at length in section 2.5. Here we content ourselves with giving a number of examples in the following exercises. These exercises should convince the reader of the relevance of monads for programming languages.

Exercise 1.7.2 *Show that the powerset functor $\mathcal{P}$ and that the list functor List each gives rise to a monad on* **Set***.*

Exercise 1.7.3 *Let S be a fixed set. Show that the functor $(\_ \times S)^S$ gives rise to a monad on* **Set***. (This monad is called the state monad,: think of S as a set of states, and that an expression's value depends on the state and that its evaluation may have side-effects on the state.)*

Exercise 1.7.4 *Let R be a fixed set. Show that the functor $R^{R^{\_}}$ gives rise to a monad on* **Set***. (This monad is called the continuation monad: think of R as a set of final results, and of functions $c : A \to R$ as continuations, or as contexts of evaluation.)*

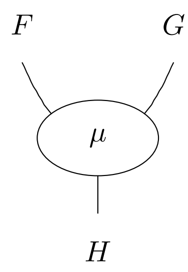
Chapter 2

String diagrams

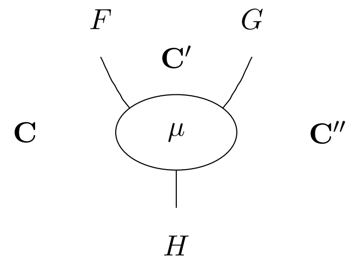
In this chapter, we introduce the graphical language of string diagrams (Section 2.1). With these new glasses we revisit adjunctions (Section 2.2), limits (Section 2.3), and equivalences of categories (Section 2.4).

2.1 String diagrams

We represent a natural tranformation $\mu : GF \to H$ as follows:



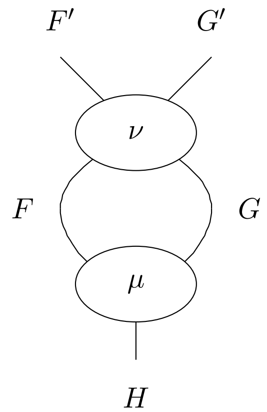
Under this representation, functors are 1-dimensional (like in the usual pasting diagrams), natural transformations are 0-dimensional (think of the circle around ρ as just a node in a graph). As for the categories, if $F : \mathbf{C} \to \mathbf{C}'$, $G : \mathbf{C}' \to \mathbf{C}''$, and $H : \mathbf{C} \to \mathbf{C}''$, then, seeing the edges of the graph as half-lines, the above figure delineates twree regions, corresponding to the three categories. In other words, in this representation, categories are 2-dimensional:



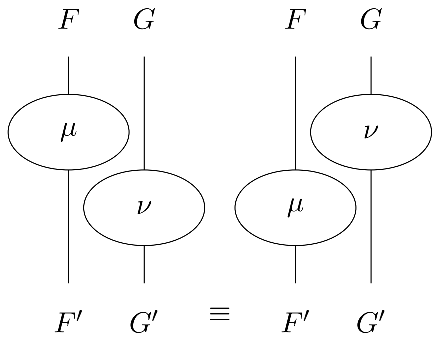
The situation is thus Poincaré dual to that of pasting diagrams:

	categories	functors	natural transformations
pasting diagrams	0	1	2
string diagrams	2	1	0

We represent the vertical composition of μ with a natural transformation $\nu : G'F' \to GF$ as follows:

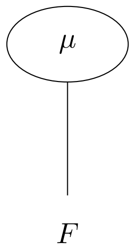


With this notation, Godement's rule amounts to the following identity:

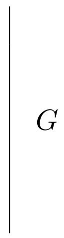


and that is the key quality of this notation: we have exchanged naturality diagrams (which are often a bit boring to check) against the ability to freely move up and down parts of drawings, just like elevators, as long as they circulate in non-overlapping cages. Such diagrams are called string diagrams.

String diagrams deal with identity functors and natural transformations implicity. We represent, say, $\mu : id \to F$ (with $F : \mathbf{C} \to \mathbf{C}$), and $id : G \to G$ (with $G : \mathbf{C} \to \mathbf{C}'$) as

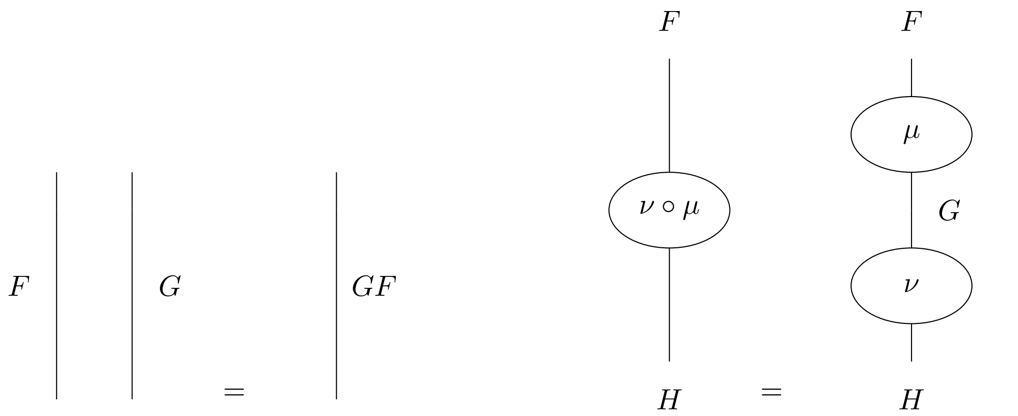


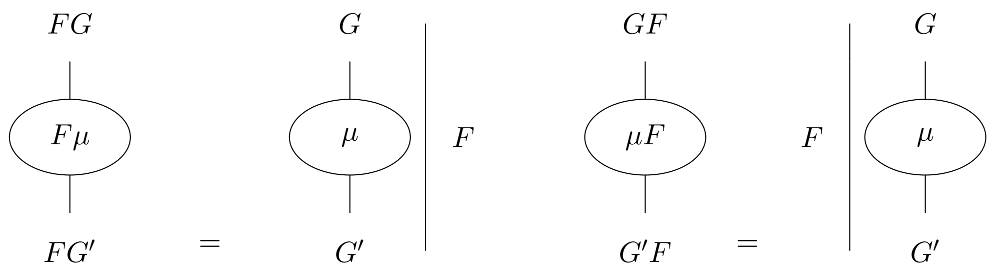
and



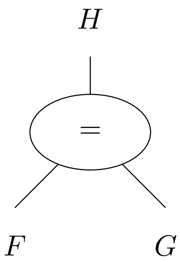
respectively.

Note that our diagrams are still annotated with the usual textual syntax for functors and natural transformations. We must therefore include the relevant commutation rules between the graphical representation and the textual one:

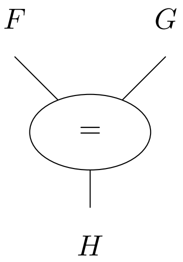




Sometimes, it will be useful to add explicit coercions (with $H = GF$)



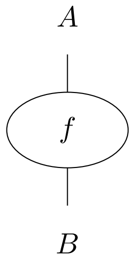
and



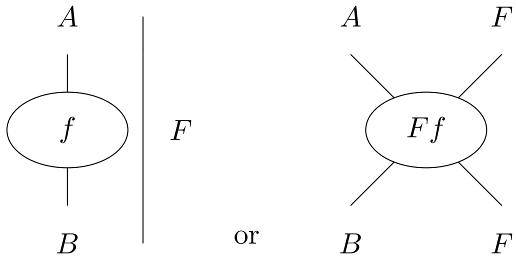
between the two sides ot the top left equality above (more on this in Section 2.3).

We ask these coercions to satisfy some equations (see Section 2.3.2).

We can also use string diagrams to describe morphisms $f : A \to B$ in a category $\mathbf{C}$. It suffices to see A and B as functors from the terminal category $\mathbf{1}$ to $\mathbf{C}$, yielding



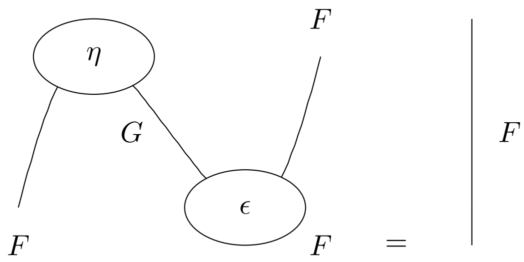
with the left and right half plane corresponding to $\mathbf{1}$ and $\mathbf{C}$, respectively. And for Ff, we can write indifferently (cf. above)



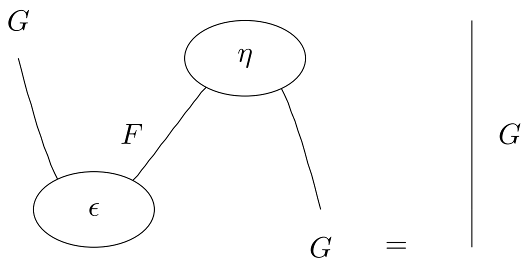
2.2 Adjunctions

The following two equations express the definition of adjunction $F \dashv G$ with unit η and counit ϵ.

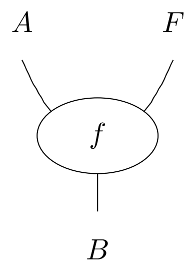
EQUATION $\eta - \epsilon$:



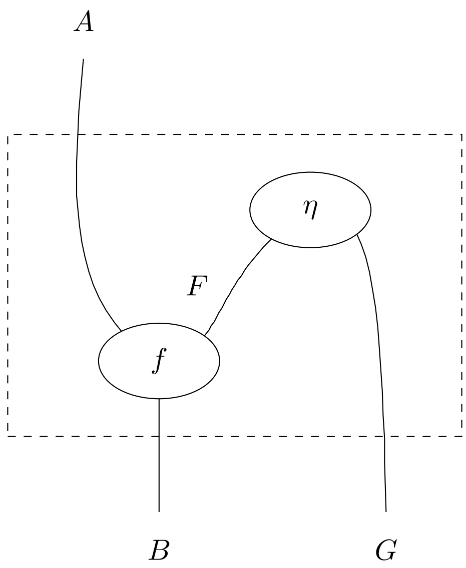
EQUATION $\epsilon - \eta$:



We now show how this definition allows us to induce the definition of adjunction in terms of the natural bijections between $\mathbf{C}[FA, B]$ and $\mathbf{C}'[A, GB]$. Given

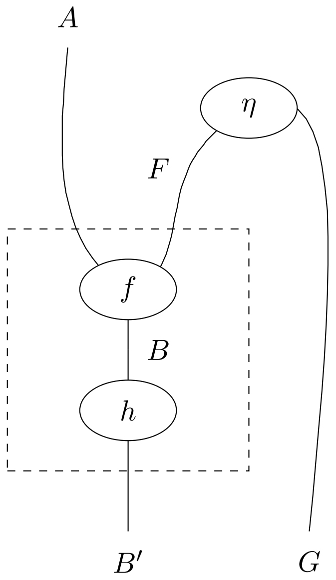


we define $\zeta(f)$ as follows:

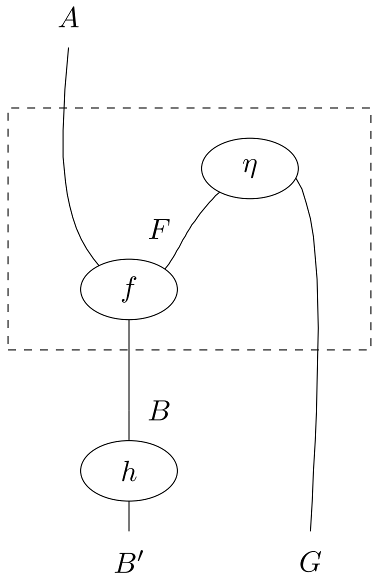


which by forgetting the details in the box is an arrow from A to GB as required.

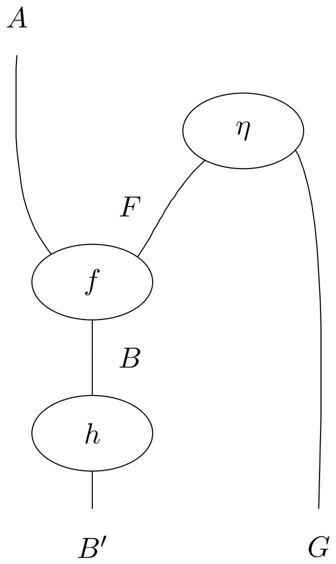
The naturality of ζ is straightforward by construction. For example $\zeta(h \circ f)$ and $Gh \circ \zeta(f)$ are obtained respectively as



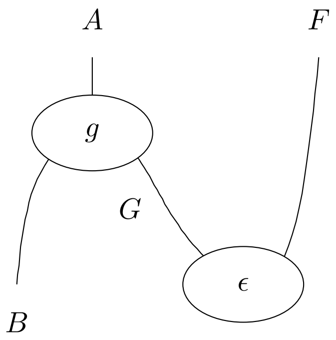
and



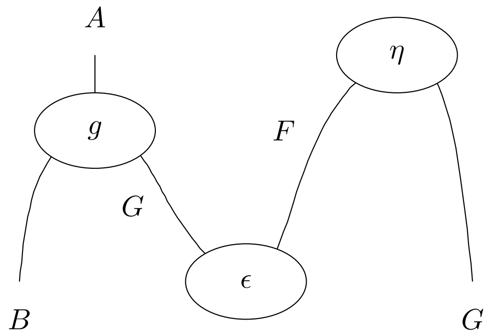
which coincide once we remove the boxes highlighting the respective constructions (or "sequentialisations", to borrow a terminology from linear logic):



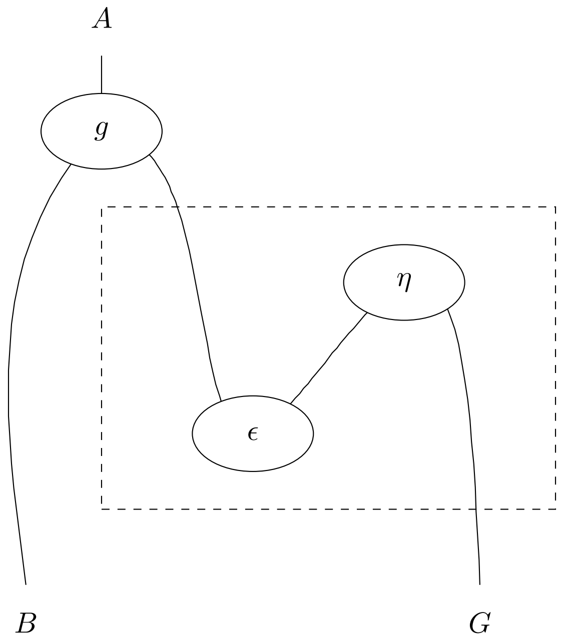
Conversely, given $g : A \to GB$, we construct $\xi(g)$ as follows:



The functions ζ and ξ are inverse. For example, let us draw $\zeta(\xi(g))$:

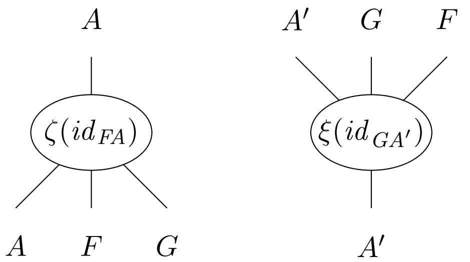


We can then move up g (Godement's rule), which allows us to isolate an $(\epsilon - \eta)$ redex.



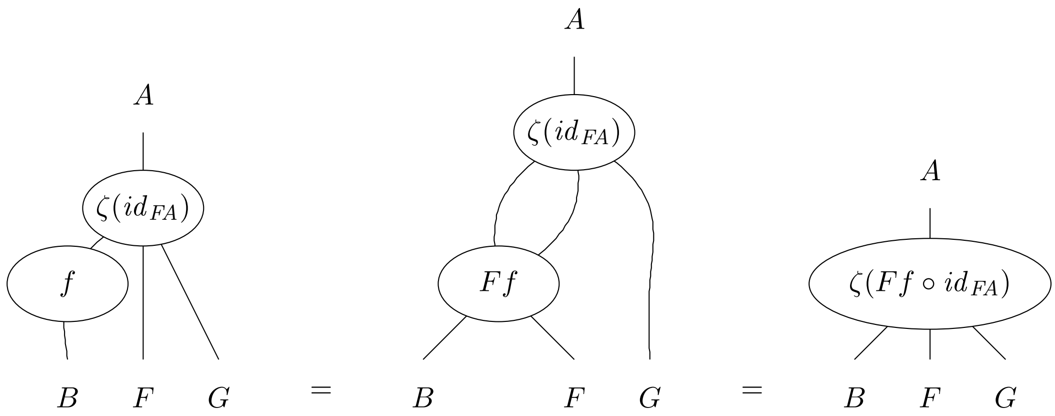
after reduction of which all what remains of the box is a wire G, i.e., we have proved $\zeta(\xi(g)) = g$.

Suppose conversely that two natural bijections ζ and ξ of the above types are given. We synthesise η_A and $\epsilon_{A'}$(on every object, thus) as follows:

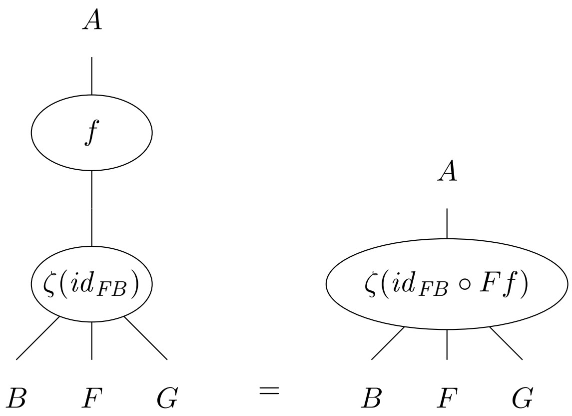


We check the naturality of η, making use of the naturality of ζ (i.e., $\zeta(f) \circ g = \zeta(f \circ Fg)$ and $Gh \circ \zeta(f) = \zeta(h \circ f)$). We work on both ends. We transform $GFf \circ \eta_A$

as follows:

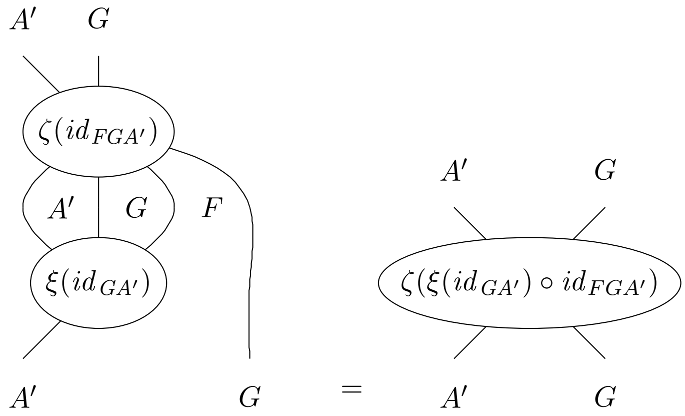


On the other hand, we transform $\eta_B \circ f$ as follows:



and we are done, since $Ff \circ id_{FA} = id_{FB} \circ Ff$.

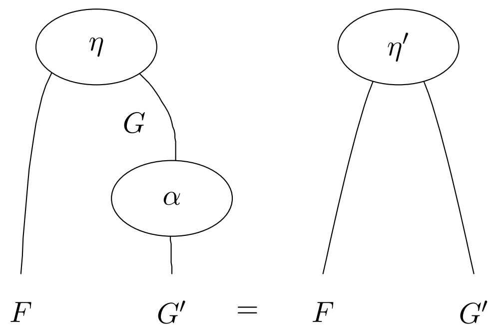
We now check Equation $(\epsilon - \eta)$ at A'. We have, by naturality of ζ:



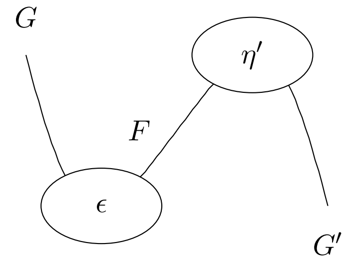
and we conclude, since $\zeta(\xi(id_{GA'})) = id$.

We conclude this section by showing the uniqueness of adjoints up to iso.

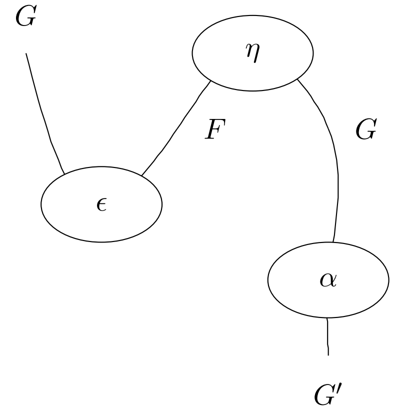
SOLUTION OF EXERCISE 1.5.7. Suppose that we have $F \dashv G$ with unit η and counit ϵ on one hand, and $F \dashv G'$ with unit η' and counit ϵ' on the other hand. Then there exists a unique iso $\alpha : G \to G'$ such that $\alpha F \circ \eta = \eta'$, or, equivalently, there exists a unique iso $\beta : G' \to G$ (the inverse of α) such that $\epsilon \circ F\beta = \epsilon'$. We synthesise α from the requirement. If



then by plugging ϵ on both sides on the F wire, we get on the right hand side

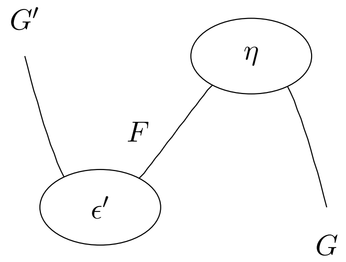


while the left hand side simplifies to α using Equality $(\epsilon - \eta)$, after having moved down α, as shown below:

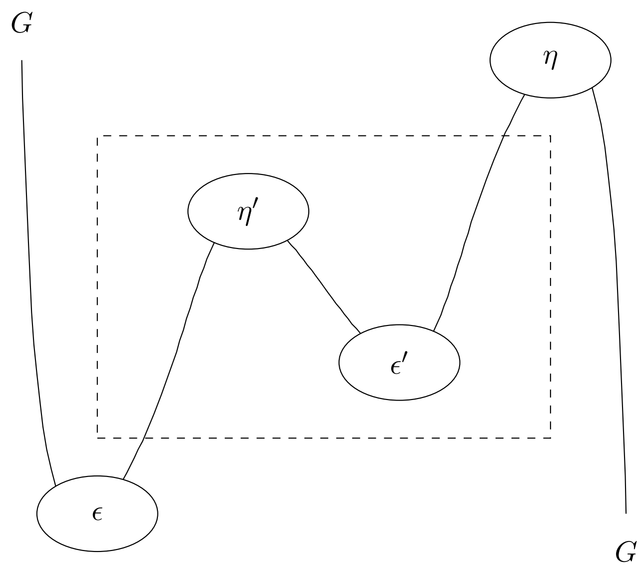


Hence α must be equal to the diagram above.

We synthesise β similarly:



It is straightforward that α (resp. β) fulfills the requirement. We check that $\beta \circ \alpha = id$. Moving down the ϵ and moving up the η we obtain

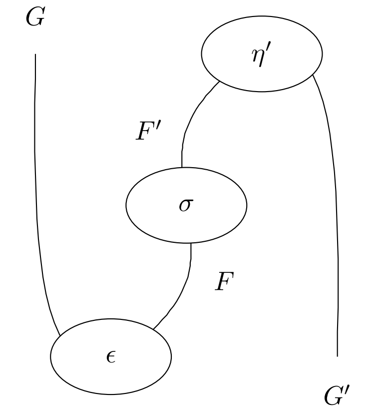


Then reducing the central $(\eta - \epsilon)$ redex leaves us with an $(\epsilon - \eta)$ redex, whose reduction in turn leaves us with a G wire, i.e. the identity natural transformation from G to G, as claimed.

2.2.1 Relating adjunctions*

Let $F \dashv G$ and $F' \dashv G'$ be two adjunctions between the same categories $\mathbf{C}$ and $\mathbf{C}'$.. These data induce a bijective correspondence between $\mathbf{C}'^{\mathbf{C}}[F', F]$ and $\mathbf{C}^{\mathbf{C}'}[G, G']$ that maps σ :

$F' \to F$ to



This natural transformation is called the *conjugate* of σ This correspondence is bijective (its inverse is defined in a dual way), and contravariantly functorial in the sense that if $F'' \dashv G''$ is a further adjunction between $\mathbf{C}$ and $\mathbf{C}'$, then the conjugate of $\sigma \circ \sigma'$ is the inversed composition of the conjugates of σ' and σ.

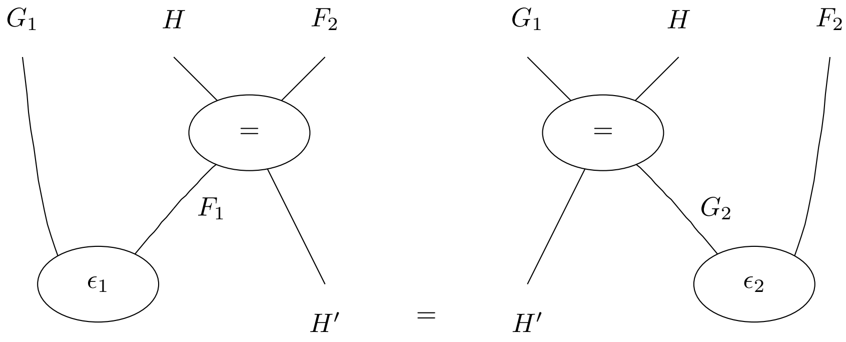
We illustrate this by proving a result on parameterised adjunctions. Let $F : \mathbf{C}'' \times \mathbf{C} \to \mathbf{C}'$ be a bifunctor, and suppose that for all A'', $\lambda x.F(A'', x)$ (a functor from $\mathbf{C}$ to $\mathbf{C}'$) has a right adjoint $G_{A''}$. We shall prove that the family formed by the $G_{A''}$'s induces a functor $G : \mathbf{C}'' \times \mathbf{C}' \to \mathbf{C}$. We describe this functor in curried form (and we also use F in curried form):

- $GA'' = G_{A''}$,
- Gf is the conjugate of Ff.

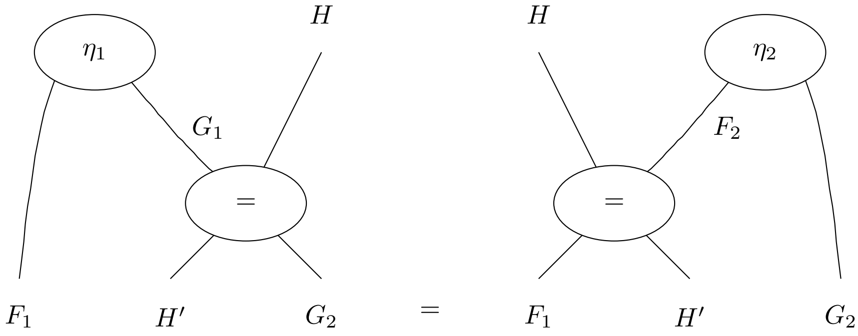
We check that this is correctly typed. If $f : A'' \to B''$, then we have $Ff : FA'' \to FB''$ (where, say, FA'' is what we previously wrote without abuse of notation as $\lambda x.F(A'', x)$), and hence $Gf : GB'' \to GA''$. So we have defined a contravariant functor from $\mathbf{C}''$ to $\mathbf{C}^{\mathbf{C}'}$, i.e., in uncurried form, we have defined as required a functor $G : \mathbf{C}''^{op} \times \mathbf{C}' \to \mathbf{C}$ such that by construction $\lambda x'.G(A'', x') = G_{A''}$, for all A''.

We next relate adjunctions that are not necessarily between the same categories. We define a category $\mathbf{Adj}$ whose objects are adjunctions. Let $F_1 \dashv G_1$ beween $\mathbf{C}_1$ and $\mathbf{C}'_1$, and $F_2 \dashv G_2$ beween $\mathbf{C}_2$ and $\mathbf{C}'_2$. Then $\mathbf{Adj}[(F_1 \dashv G_1), (F_2 \dashv G_2)]$ consists of the pairs of functors $H : \mathbf{C}_1 \to \mathbf{C}_2$ and $H' : \mathbf{C'}_1 \to \mathbf{C}'_2$ such that

- $F_2 H = H' F_1$,
- $HG_1 = G_2 H'$,
- and H' commutes with counities:

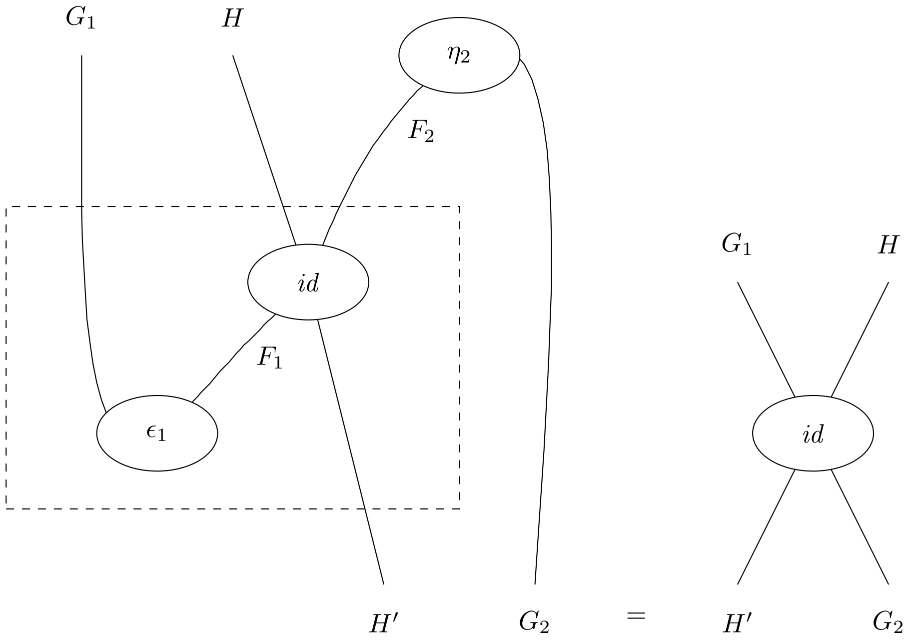


The third condition (in the presence of the first two conditions) is equivalent to the following symmetric condition of commutation with unities:



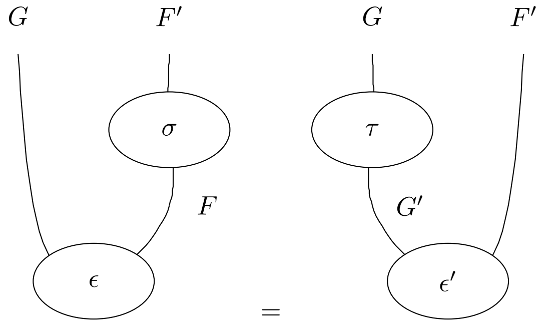
(2.1)

To establish this, we apply an $(\eta - \epsilon)$ expansion to the F_1 wire on the right. Then, pushing η_1 upwards on both sides, we are left to show:

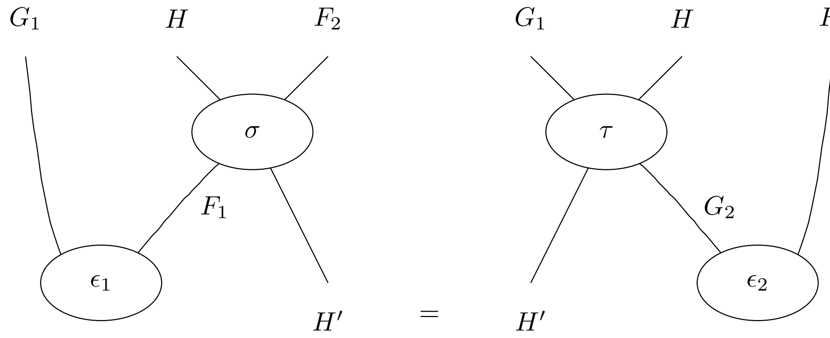


Indeed, we get from the left hand side to the right hand side by successively applying the commutation of H' with counities, pushing id up, and then reducing the rest of the diagram to a G_2 wire thanks to $(\epsilon - \eta)$.

So far, we have two notions of morphisms between adjunctions: the notion just defined, and for adjunctions between the same categories: in the latter case a morphism is a natural transformation $\sigma : F' \to F$. But it is more symmetric to look at it as the pair formed by σ and its conjugate τ, where σ and τ are mutually defined via the following equation:



There is a more general notion of morphism between two adjunctions, that has the previous ones as instances. The general definition takes as morphism from $F_1 \dashv G_1$ to $F_2 \dashv G_2$ the quadruples of the form $(H, H', \sigma : F_2H \to H'F_1, \tau : HG_1 \to G_2H')$ such that the following commutation property hoids with respect to counities:



This equality is equivalent to a dual commutation property with respect to unities (obtained by composing the two diagrams above with η_1 on the left and η_2 on the right. The conjugation instance corresponds to the case where $H = H' = id$, and that of morphism in the category **Adj** to the case where $\sigma = \tau = id$.

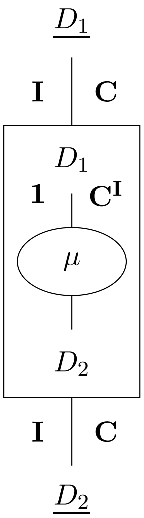
2.3 Adjunctions and limits

We devote this section to a graphical proof of Proposition 1.5.6. On the way, we shall also give a graphical solution to Exercise 1.5.9. We are facing a difficulty: given a diagram $D : \mathbf{I} \to \mathbf{C}$, how should we draw a cone of domain ΔC to D? The domain is a functor from $\mathbf{1}$ to $\mathbf{C}^{\mathbf{I}}$, hence we have no choice but to consider D also as a functor from $\mathbf{1}$ to $\mathbf{C}^{\mathbf{I}}$. But when we shall deal with the diagram FD (with $F : \mathbf{C} \to \mathbf{C}'$, we shall want D to be now a functor from $\mathbf{I}$ to $\mathbf{C}$. Under this guise, we denote it as $\underline{D}$.

Note that in any CCC, there is a bijective correspondence betwen the morphisms from A to B and the points of B^A, cf. Exercise 1.3.5. We shall use here underlining as an *explicit coercion* from the latter to the former.

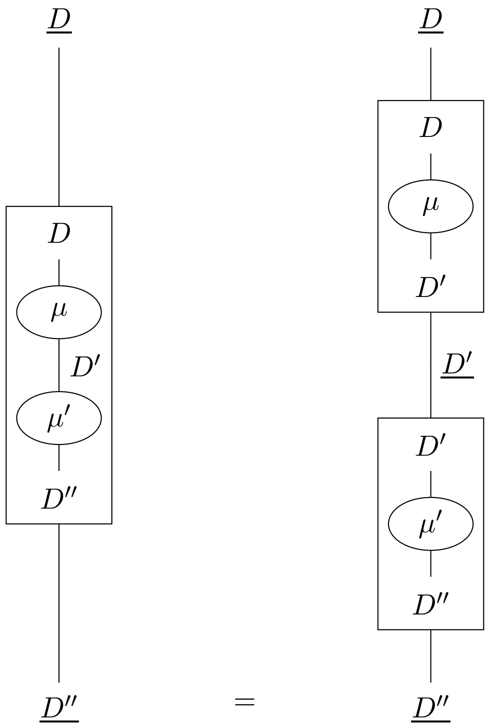
2.3.1 Coercions in string diagrams

Graphically, we shall introduce boxes of the following kind:



where the contents of the box is a string diagram living in $\mathbf{Cat}[\mathbf{1}, \mathbf{C}^{\mathbf{I}}]$ while the whole diagram, once coerced, lives in $\mathbf{Cat}[\mathbf{I}, \mathbf{C}]$, and can be inserted in a larger diagram (e.g. by placing a wire $F : \mathbf{C} \to \mathbf{C}'$ on the right.

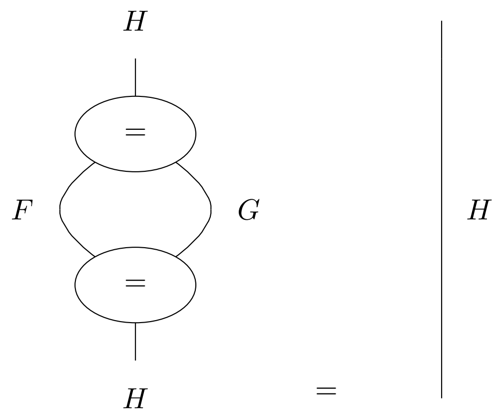
We have the following law of commutation between coercion and composition:



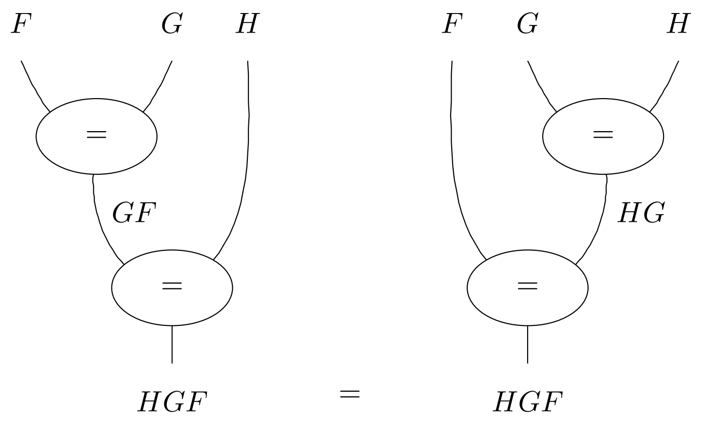
(2.2)

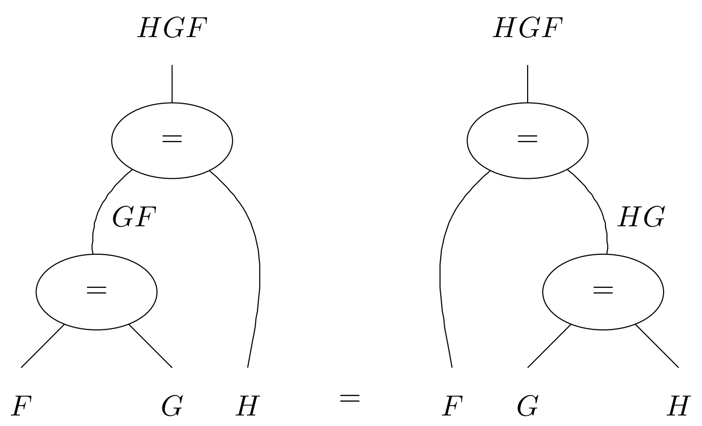
2.3.2 The laws of explicit equality

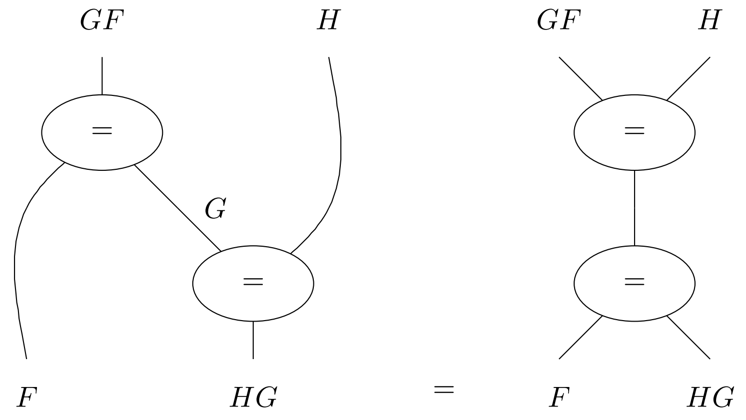
We pause here to resume our discussion of explicit equality nodes in string diagrams. We axiomatize them through the following equalities.



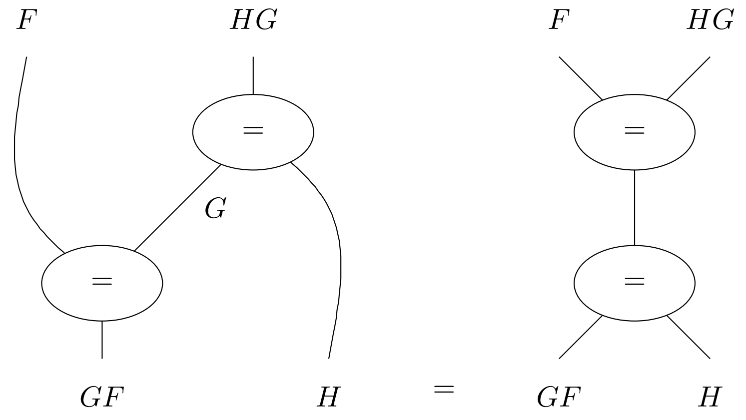
(2.3)





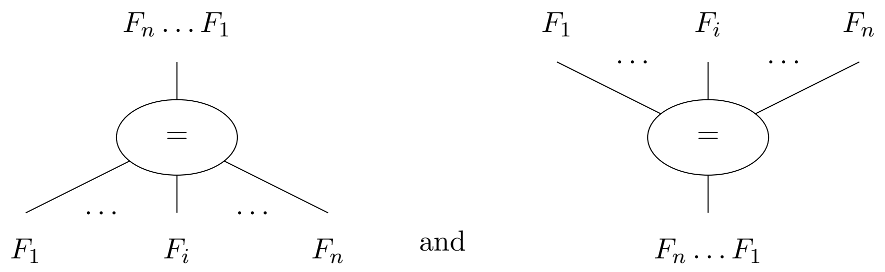


(2.4)

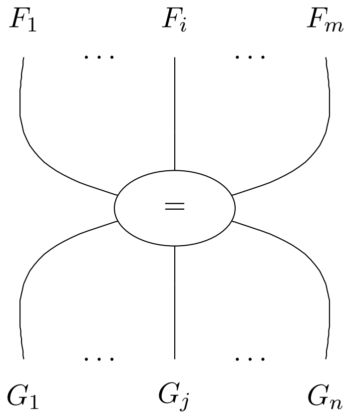


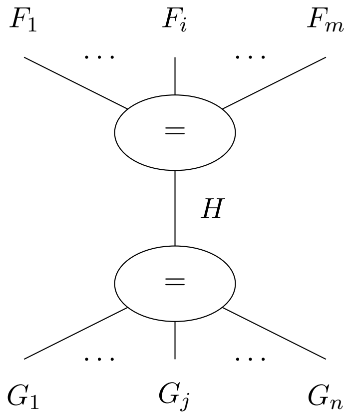
(2.5)

The associativity equation allows us to define unambiguously equality nodes of arbitrary arities:



And finally, when $F_m \dots F_1 = G_n \dots G_1$ $(= H)$, we shall write



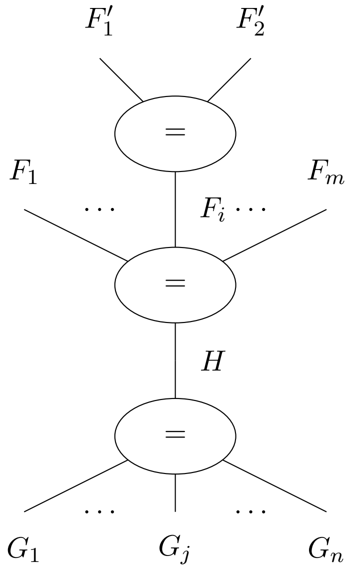


as an abbreviation of

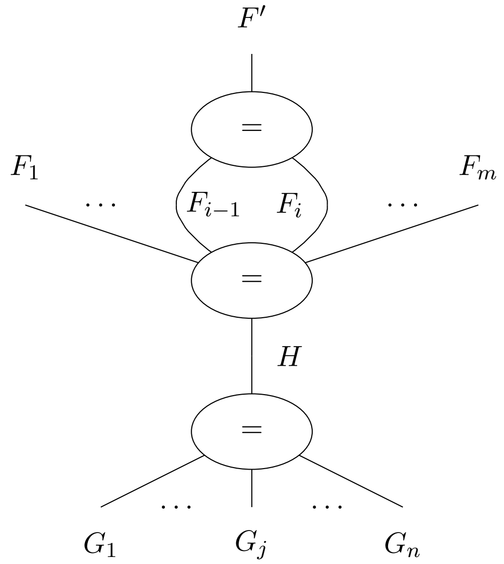
(2.6)

Our set of equations for the equality nodes has the following completeness, or coherence property: any connected string diagram with m input wires and n input wires that is written only with equality nodes is provably equal to to a single wire when $m = n = 1$, and to some diagram of the form (2.6) otherwise. We shall call such a diagram an equality normal form. In the proof, we tacitly assume that the notation (2.6) just means the wire H when $m = n = 1$. The proof is by induction on the size of the diagram. Let us push upwards one of the top equality nodes in the diagram. Applying induction to the rest, we are in one of the following four situations:

1.

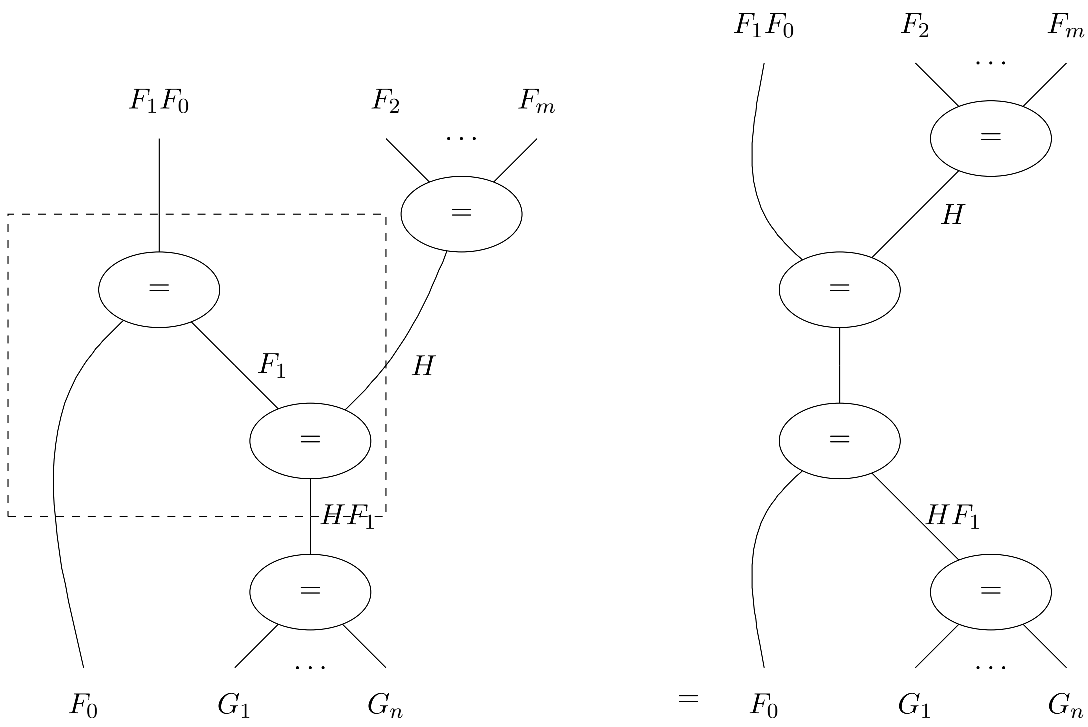


In this case, there is nothing to do, since the whole diagram is also in equality normal form.

2. 

In this case, we conclude using Equation (2.3). Note that if $m = 2$ and $n = 1$, we get a single wire as the result.

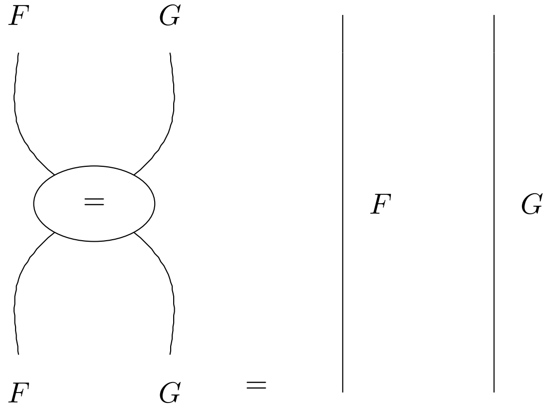
3. The chosen top operator is as in case (2), but $i = 1$. Then we conclude using Equation (2.4):



(whrere $H = F_n \dots F_2$).

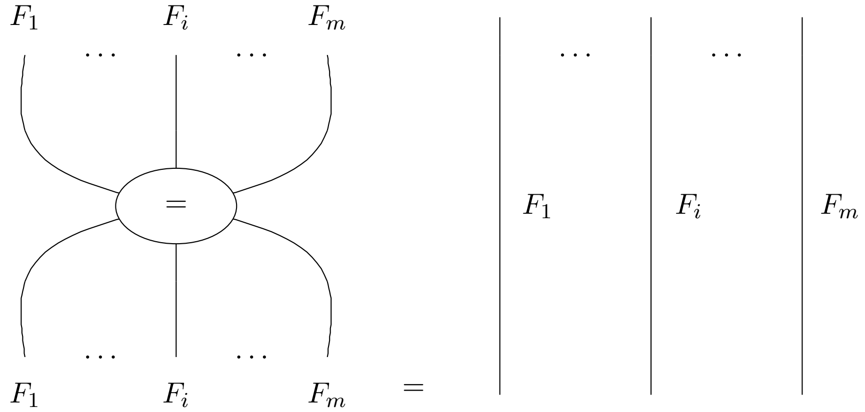
4. The chosen top operator is as in case (2), but $i - 1 = m$. We apply Equation (2.5).

But this is not yet enough. We also need a rule that allows us to remove an equality node when it relates a sequence $F_1, \dots, F_m$ to the identical sequence $F_1, \dots, F_m$. So we also require the converse of Equation 2.3:



(2.7)

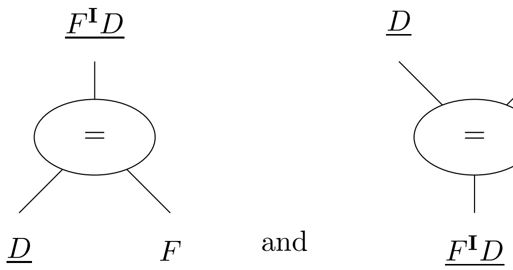
By associativity and by induction, this equality extends easily to sequences of functors:



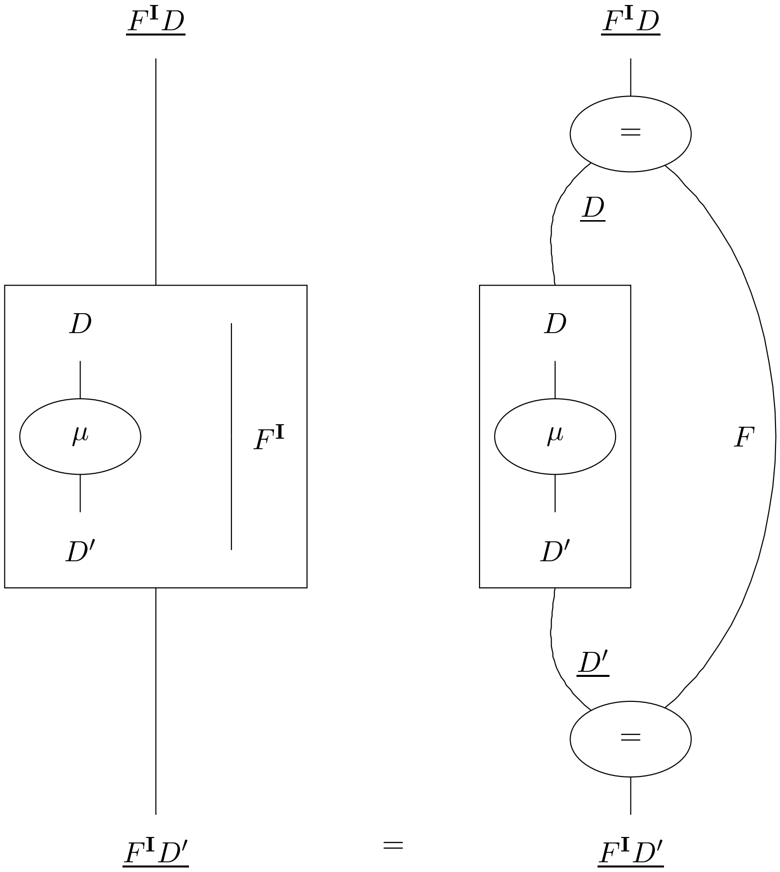
(2.8)

2.3.3 Coercions in action

We can now go ahead with the subject of this section. The action of the functor $F^{\mathbf{I}}$ is described on objects by the equality $\underline{F^{\mathbf{I}} D} = F\underline{D}$ and thus graphically by the following operators:



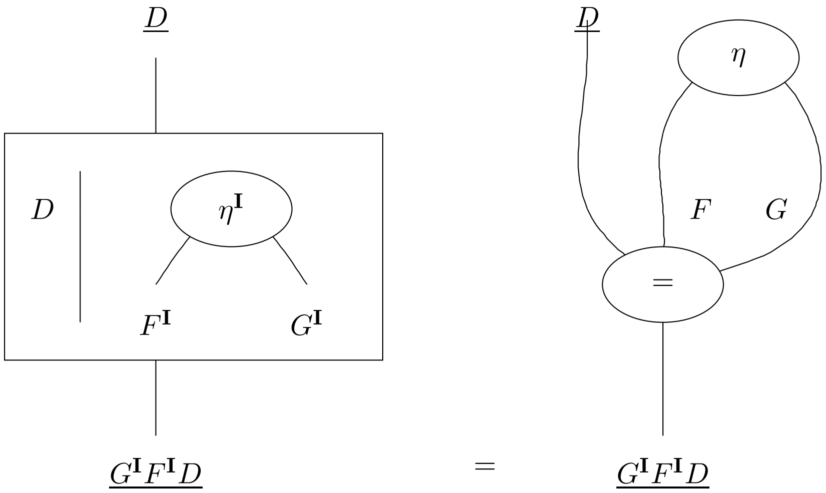
and on morphisms by the following equation:



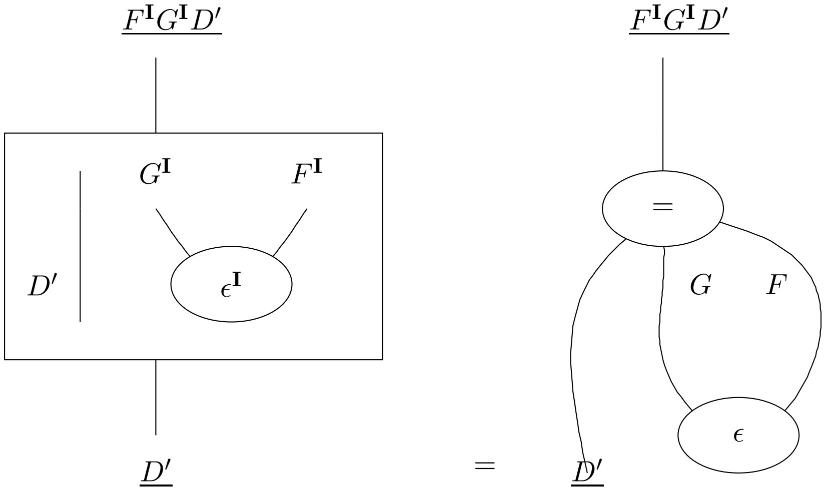
(2.9)

Solution of Exercise 1.5.9. We have to prove that if $F \dashv G$ (with $F : \mathbf{C} \to \mathbf{C}'$), and if $\mathbf{I}$ is a graph, then $F^{\mathbf{I}} \dashv G^{\mathbf{I}}$.

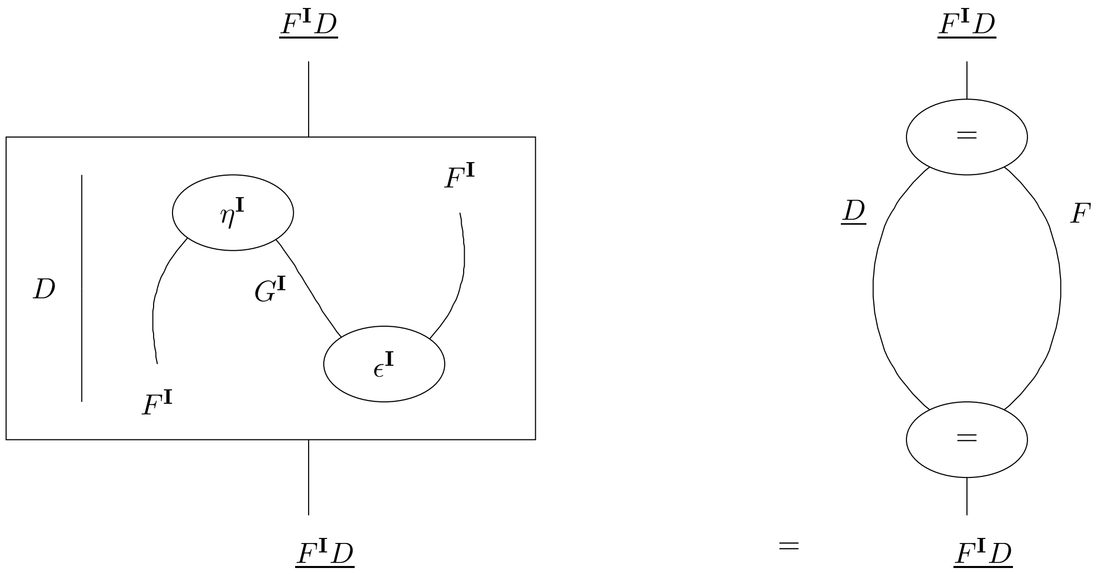
We define $\eta^{\mathbf{I}}$ as follows:



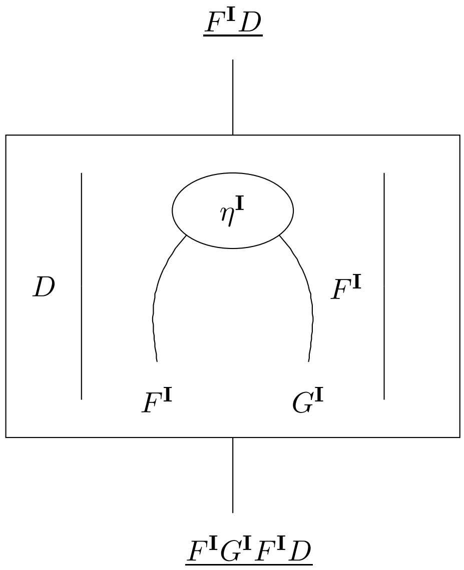
We define $\epsilon^{\mathbf{I}}$ similarly:



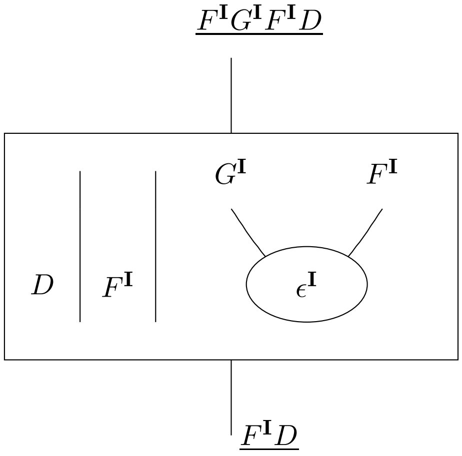
We show, say, that Equation $(\eta - \epsilon)$ holds for $\eta^{\mathbf{I}}$ and $\epsilon^{\mathbf{I}}$. We have to prove



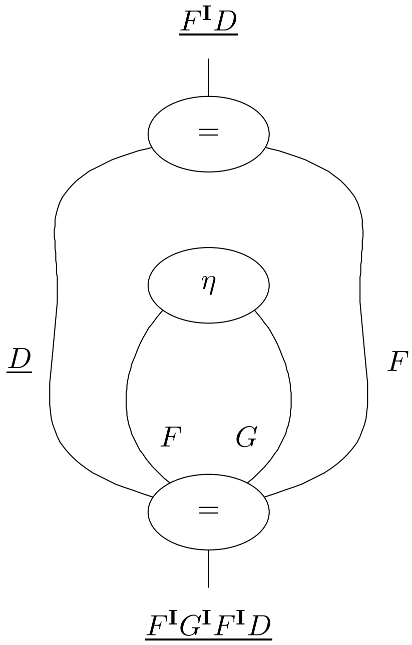
Using equation (2.2), we can decompose the left hand side as the vertical compostion of



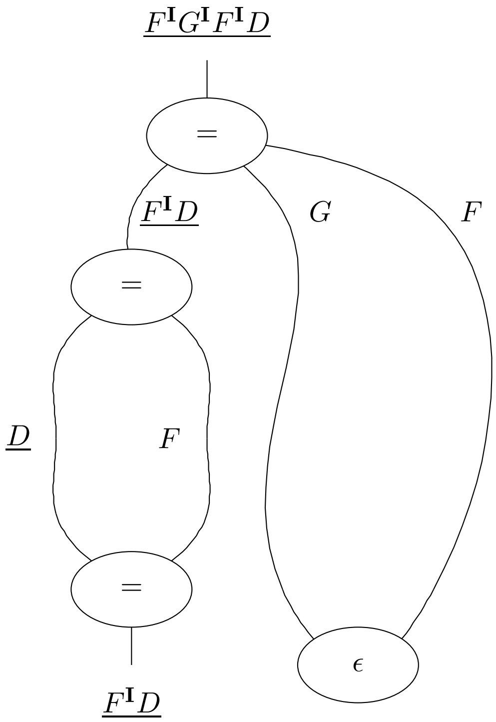
and



Applying equation (2.9) and then unfolding the definition of $\eta^{\mathbf{I}}$ in the top diagram, we get



Applying the definition of $\epsilon^{\mathbf{I}}$, the bottom diagram becomes



and we conclude using Equations 2.8 and $(\eta - \epsilon)$. □

We now turn to limit preservation. We first state two equations that are satisfied by the functor Δ:

$$G^{\mathbf{I}}\Delta' = \Delta G \qquad \text{and} \qquad \underline{\Delta\mu_{A'}} = \mu\underline{\Delta' A'}$$

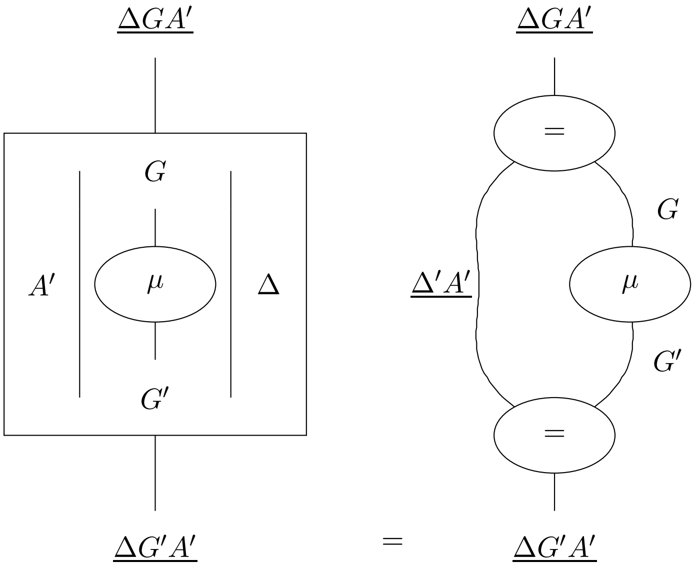
We refer to the discussion following Proposition 1.5.6 for a proof of the first equality (on objects). For the second equality, we have (on objects), for every $i : \mathbf{I}$:

$$\underline{\Delta\mu_{A'}}_i = \mu_{A'} = \mu_{\underline{\Delta' A'}i} = (\mu\underline{\Delta' A'})_i$$

We also observe that we have $\underline{\Delta G A'} = G\underline{\Delta' A'}$ as a special case of the second equality; it can also be obtained from the first one and from the equality $\underline{F^I D} = F\underline{D}$, as follows:

$$\underline{\Delta G A'} = \underline{G^I \Delta' A'} = G\underline{\Delta' A'}$$

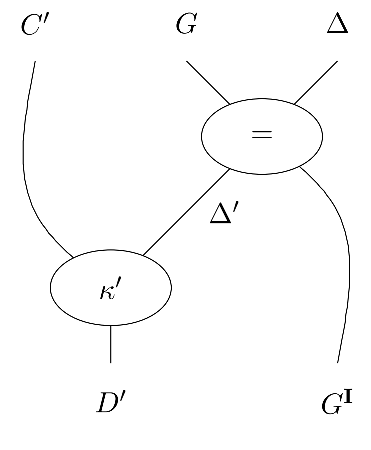
Somehow, the equation $\underline{\Delta\mu_{A'}} = \mu\underline{\Delta' A'}$ says that μ survives safely the intermediate passage through $G^{\mathbf{I}}$ (cf. the definition of $\eta^{\mathbf{I}}$ and $\epsilon^{\mathbf{I}}$ above). Graphically, this equation is expressed as follows:



(2.10)

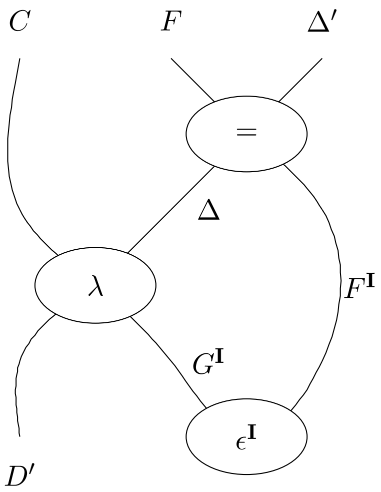
Proof of Proposition 1.5.6.

Let $F \dashv G$ be an adjunction between two categories $\mathbf{C}$ and $\mathbf{C}'$, let $D' : \mathbf{I} \to \mathbf{C}'$ be a diagram that has a limiting cone $\kappa' : \Delta' C \to D'$. We want to show that the following cone κ from GC' to GD' (which we must write here a $G^{\mathbf{I}}D'$, since D' is treated as going from $\mathbf{1}$ to $\mathbf{C}^{\mathbf{I}}$, just as $\Delta' C'$), is a limiting cone.

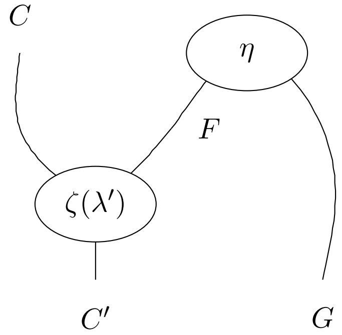


(κ is a cone by construction, since a string diagram denotes a natural transformation).

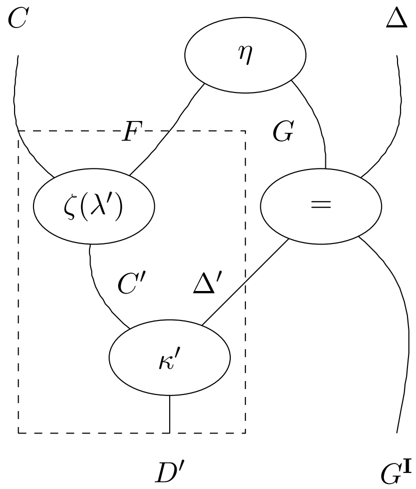
Consider next an arbitrary cone $\lambda : \Delta C \to G^{\mathbf{I}}D'$. We look for a mediating arrow from C to GC'. For this we use λ to construct a cone λ' over D', as follows:



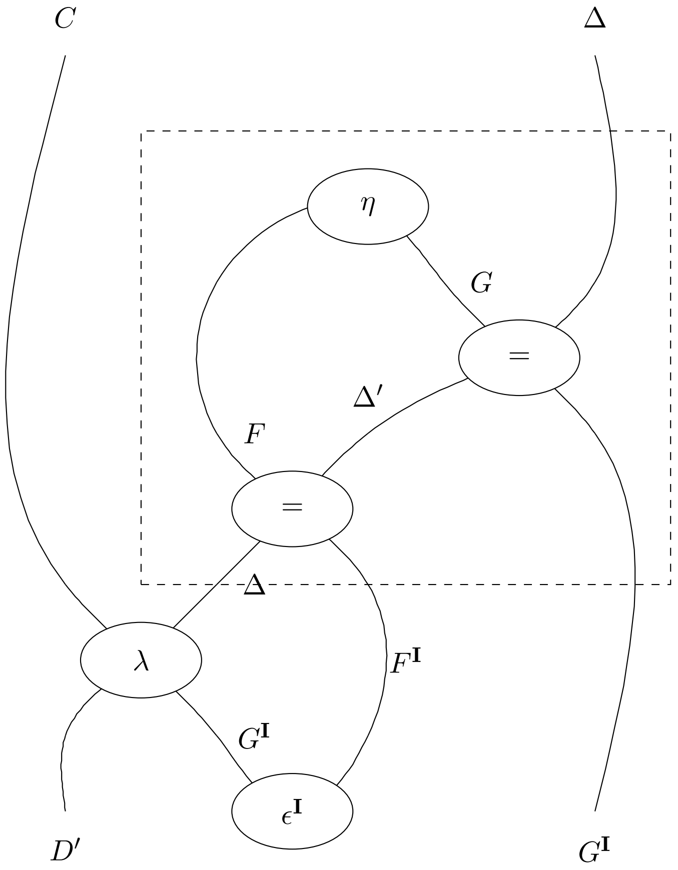
and then we get our candidate mediating arrow $\zeta(\lambda)$ as follows:



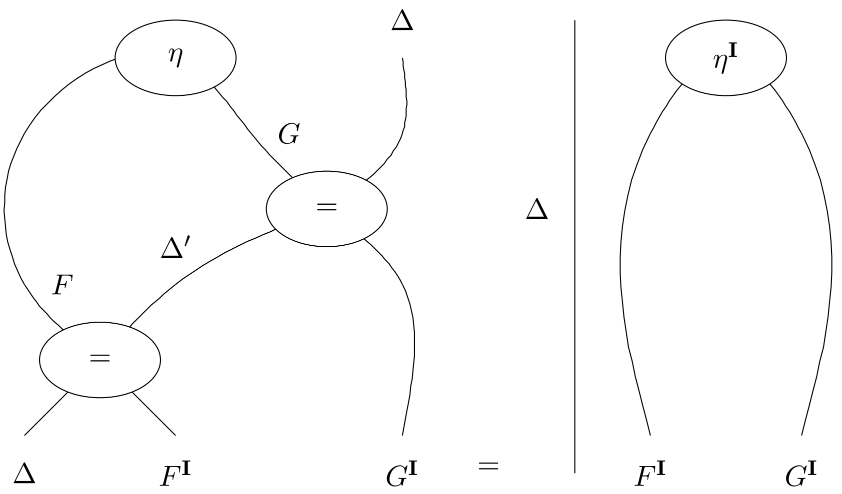
We have to check that $\kappa \circ \Delta\zeta(\lambda) = \lambda$. The left hand side of this equality is the following diagram:



where we have isolated a subregion which can be replaced by λ', since $\zeta'(\lambda')$ is the mediating arrow for λ'. After replacement, the diagram is as follows:



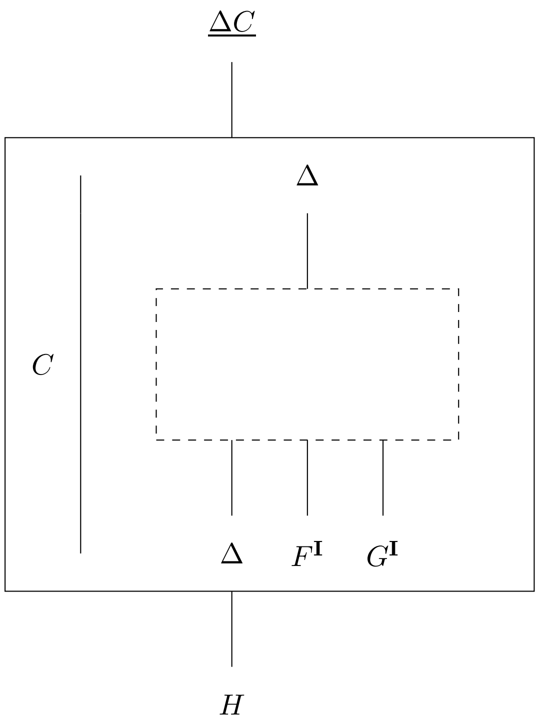
We shall prove the following equality:



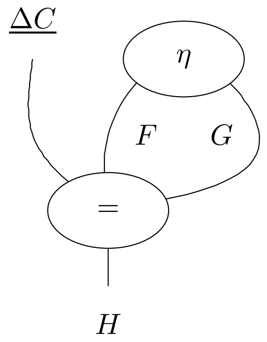
(2.11)

from which we will be able to conclude, since by using (2.11), followed by the equality $(\epsilon - \eta)$ (applied to $F^{\mathbf{I}} \dashv G^{\mathbf{I}}$), we get λ, as required.

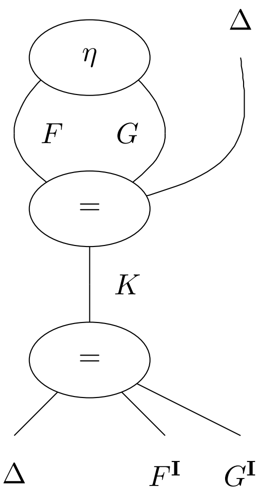
We prove (2.11) pointwise, by placing either ot the two sides of the equality in the following context (a context E is a string diagram with a hole, which we denote by a dashed box with empty content):



where $H = \underline{G^{\mathbf{I}}F^{\mathbf{I}}\Delta C} = \underline{\Delta GFC}$. In this context, we can replace $\eta^{\mathbf{I}}$ with its definition (performing an $=$ expansion of the H wire) yielding

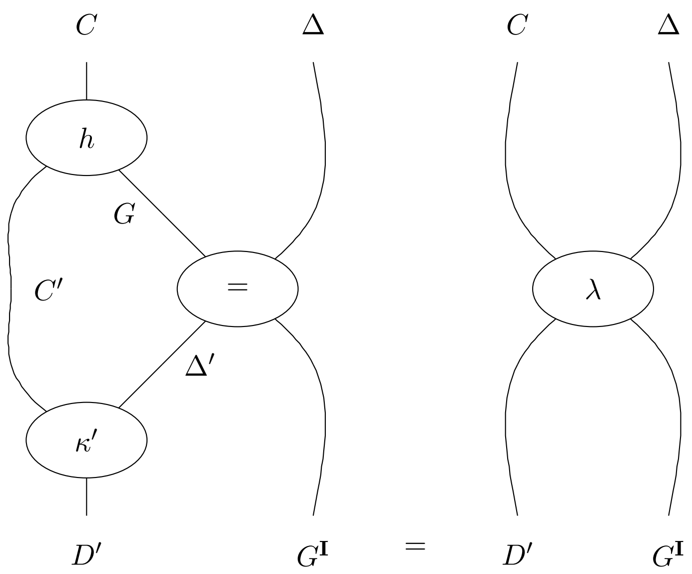


On the other hand, writing the left hand side of Equation (2.11) as

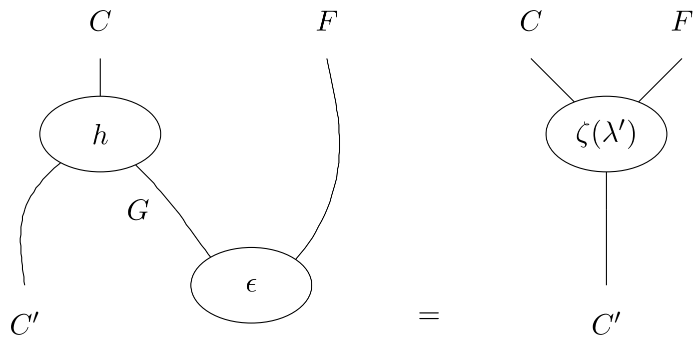


where $\underline{K} = H$, it is apparent that we arrive at the same result, applying Equation (2.10) and Equation (2.2).

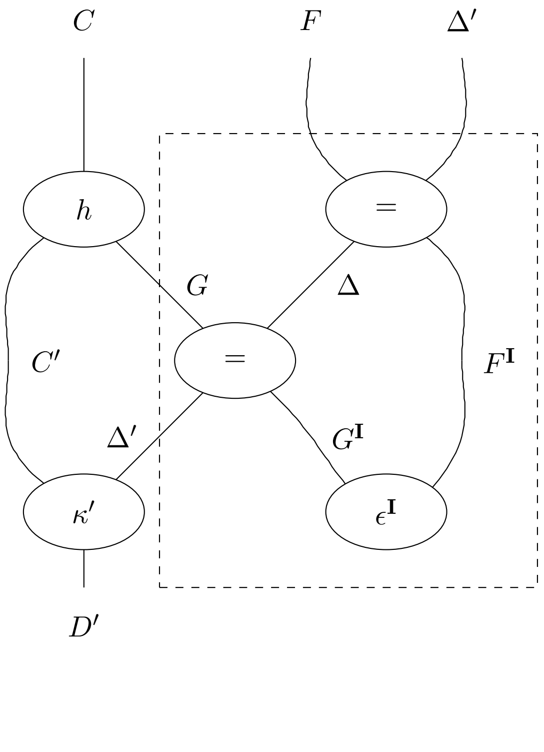
It remains to show the uniqueness of the mediating arrow, i.e., that if $h : C \to GC'$ is such that



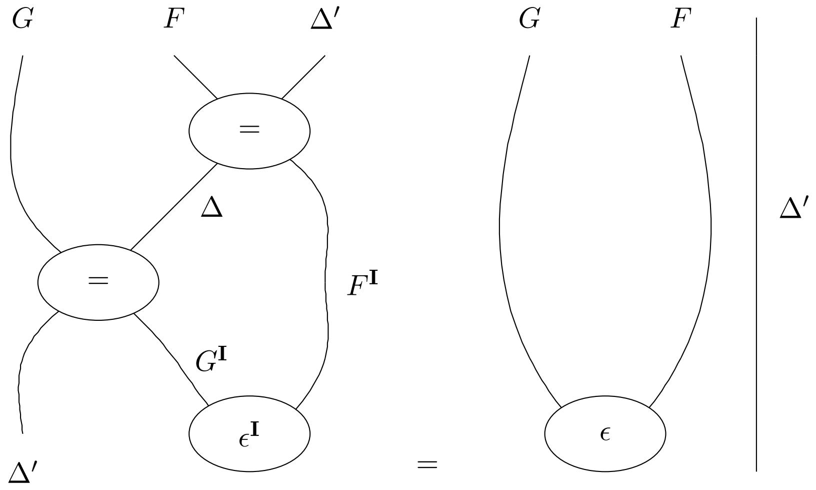
then $h = \zeta(\lambda)$. We shall prove:



from which $h = \zeta(\lambda)$ will follow by $(\epsilon - \eta)$. To prove this equality, it suffices to prove that the left hand side is mediating from λ' to κ' (since κ' is a limiting cone by assumption). Now, replacing λ in the definition of λ', we get the following string diagram for λ':



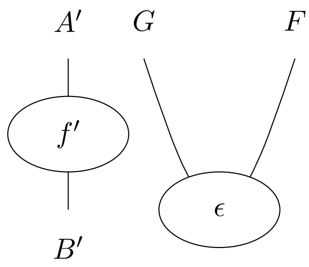
and we are done thanks to the following equality:



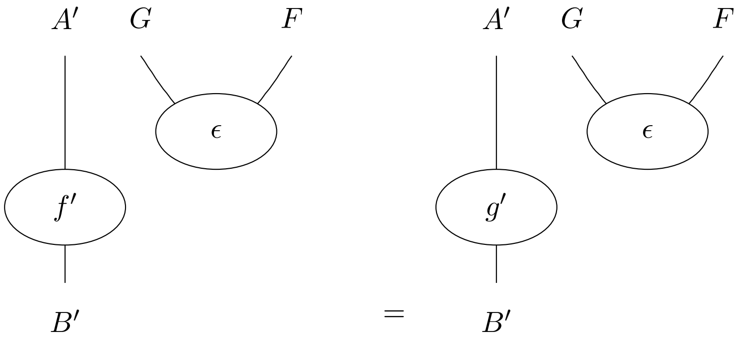
which is proved in the same way as Equation (2.11).

2.4 Equivalences of categories

In this section, we provide proofs for the two propositions stated in Section 1.6. PROOF OF PROPOSITION 1.6.1. We recall that we are given an adjunction $F \dashv G$. We have to prove that: (1) G is faithful if and only if every component of the counit is an epi, (2) G is full if and only if every component of the counit is a split mono, and (3) G is full and faithful if and only if the counit is iso. (1) Let us suppose first that each component of ϵ is epi, and suppose that $Gf' = Gg'$ (for $f', g' : A' \to B'$). Then we can replace Gf' with Gg' in

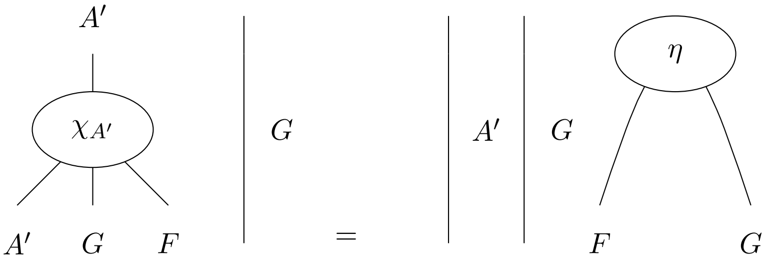


Pushing up ϵ, we get thus

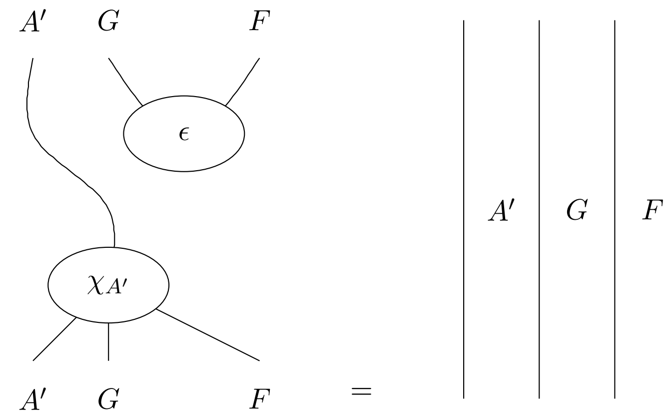


and we conclude $f' = g'$ from the assumption. Suppose conversely that G is faithful and that we have $f'\epsilon_{A'} = g'\epsilon_{A'}$. Then by plugging η on top of both sides of the equality, and applying $(\epsilon - \eta)$, we get $Gf' = Gg'$, and hence $f' = g'$.

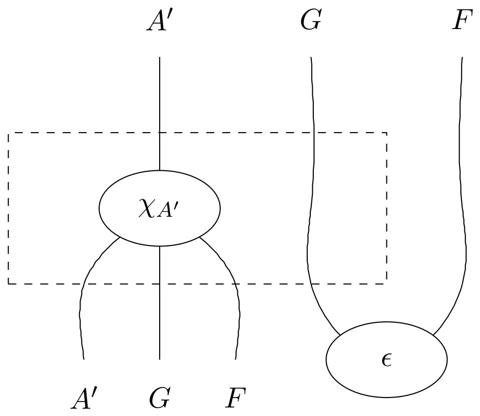
(2) Suppose first that G is full. We are looking for a family of morphisms $\chi_{A'} : A' \to FGA'$ such that $\chi_{A'} \circ \epsilon_{A'} = id_{FGA'}$. Since G is full, we can first look for a morphism from GA' to $GFGA'$, and one tempting such morphism is $\eta_{GA'}$. We thus take for $\chi_{A'}$ a morphism such that



We have to prove:

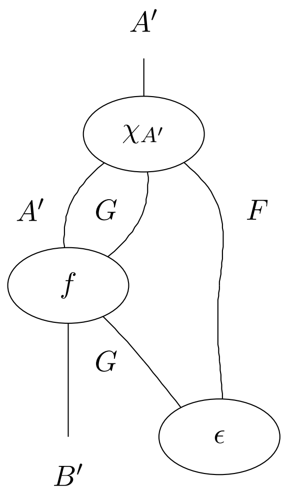


This is achieved by moving ϵ down (and to the right), so as to be able to highlight $G\chi_{A'}$:

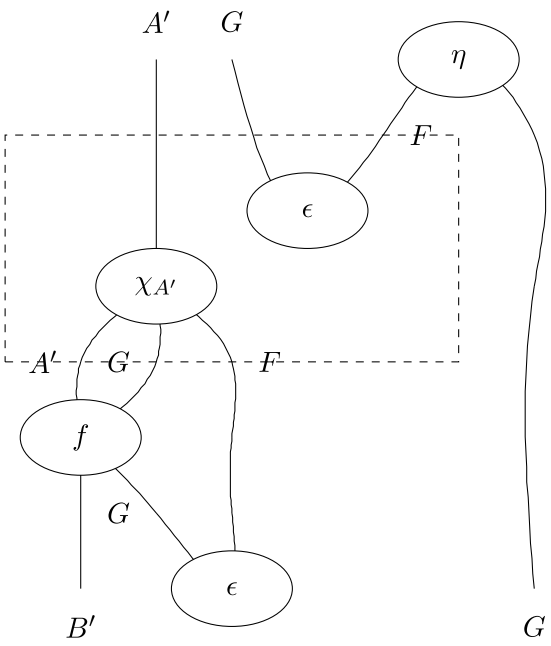


from where the conclusion is easy, replacing $G\chi_{A'}$ with its definition and applying $(\eta - \epsilon)$.

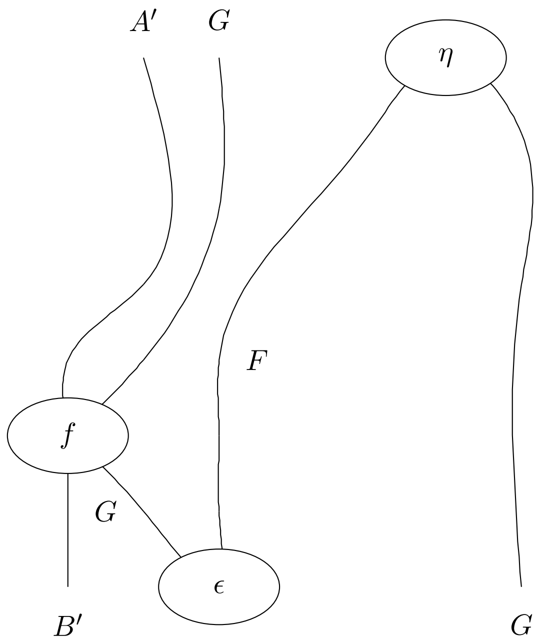
Conversely, suppose that we have a family of morphisms $\chi_{A'}$, each left inverse for $\epsilon_{A'}$. Let $f : GA' \to GB'$. We are looking for a morphism from A' to B', depending on f and whose definition makes use of $\chi_{A'}$. A natural candidate is



We have tho check that, adding a G wire on the right, we recover f. We in fact add an $(\epsilon - \eta)$ expansion of G so as to prepare the ground for the elimination of $\chi_{A'}$:



Now we apply our assumption and replace the region by three wires:

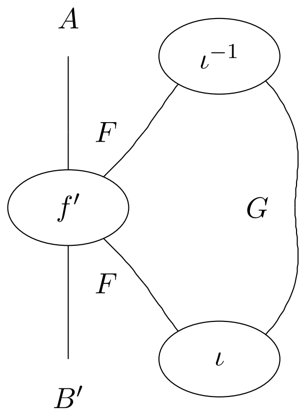


and we get f as wished, after pushing f up and applying $(\epsilon - \eta)$.

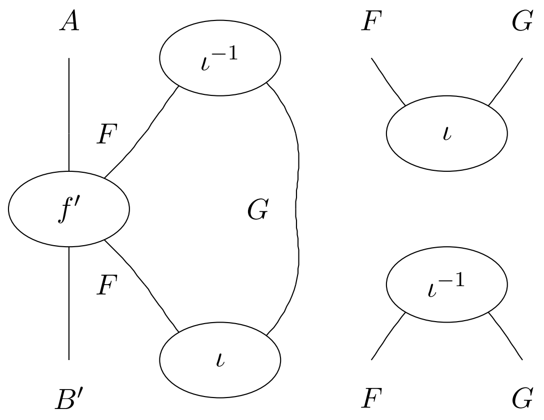
(3) The last part of the statement is an obvious consequence of the first two parts, keeping in mind that a natural transformation is an iso if and only if all its components are iso (cf. Exercise 1.3.9). $\square$

Proof of Proposition 1.6.2. We recall that, given a functor $F : \mathbf{C} \to \mathbf{C}'$, we have to prove that the following properties are equivalent: (1) there exists a functor $G : \mathbf{C}' \to \mathbf{C}$ and two natural equivalences $\iota : GF \to id_{\mathbf{C}}$ and $\iota' : FG \to_{\mathbf{C}'}$, (2) F is part of an adjunction $F \dashv G$ in which the unit and the counit are natural isomorphisms, and (3) F is full and faithful and $\forall C' : \mathbf{C}' \ \exists C : \mathbf{C} \ (C' \cong FC)$.

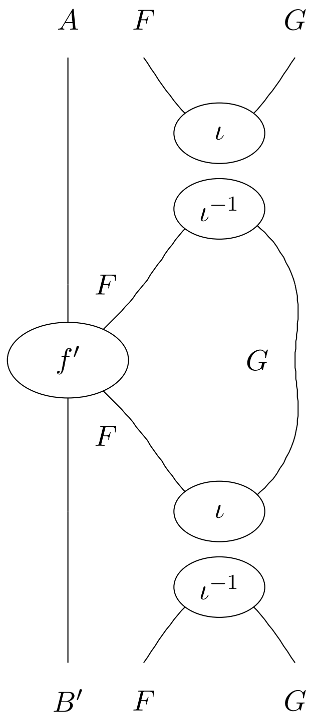
Obviously, (2) implies (1). We show that (1) implies (3). For C, take GC'. To prove the faithfulness of G in Proposition 1.6.1, we used only the existence of a natural transformation $\epsilon : FG \to id$ whose components are epi. We might thus as well have used the transformation ι'. So we note that (1) implies that G is faithful, a fact that we shall exploit to prove the fullness of F. Using likewise that all components af ι^{-1} are (a fortiori) mono, we get that also F is faithful. We now prove that F is full,. Let $f' : FA \to FB$. Following the steps taken in the proof of Proposition 1.6.1, a natural candidate for a preimage f is



We cannot proceed directly to prove $Ff = f'$. But we know that G is faithful, so it suffices to prove $GFf = Gf'$. Making once more use of the inverse isomorphisms ι and ι', we can display GFf as



We can then rearrange the picture so as to make two new cancellations of inverses visible:



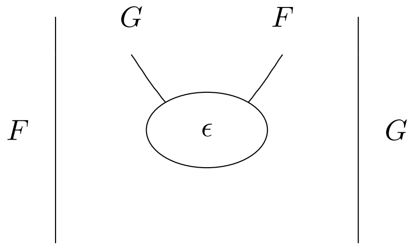
and we rare done, i.e. after reducing the top and bottom inverse isomorphisms, we get Gf' as required.

We prove finally that (3) implies (2). We set $GC' = C$, and we call $\epsilon_{C'}$ the iso given in the statement. Let $f : FC \to C'$. We have to find $g : C \to GC'$ such that $\epsilon_{C'} \circ Fg = f$, i.e., such that $Fg = \epsilon_{C'}^{-1} \circ f$. Such an arrow exists and is unique since F is full and faithful. We have thus an adjunction. The fact that F is full and faithful then also implies that all components of the unit of this adjunction are iso. This completes the proof.

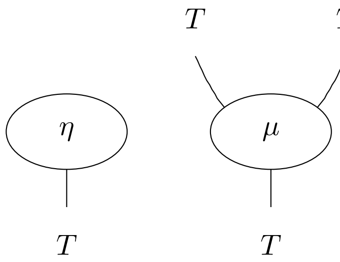
(Remark: one may be curious of a direct proof of (1)⇒(2): one can take ι' as ϵ, but then an attempt to define η directly as a string diagram in terms of ι, ι' and their inverses does not lead us anywhere; one has instead to establish first that F is full and faithful, and from there we can define η_C as the unique preimage of ι'^{-1}_{FC}; thus, the above organisation of the proof, that we borrowed from [36], is the right one.) □

2.5 Monads

We recall from Section 1.7 that, given an adjunction $F \dashv G$ between two categories $\mathbf{C}$ and $\mathbf{C}'$, we can focus on $\mathbf{C}$ and consider the endofunctor $T = GF : \mathbf{C} \to \mathbf{C}$. We then read the unit as $\eta : id \to T$, and, by placing an F on the left and a G on the right, we get a natural transformation $\mu : TT \to T$:

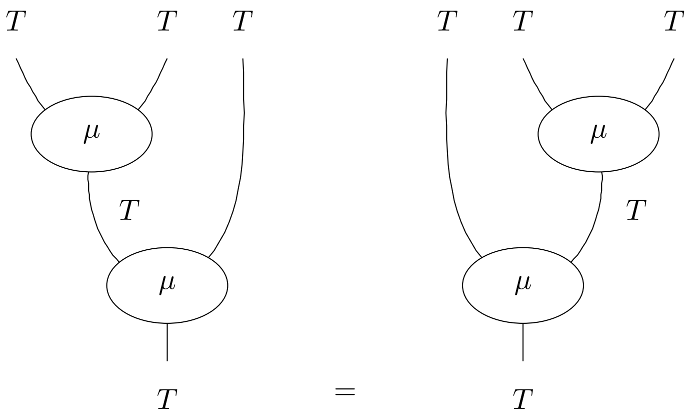


We also recall that a monad is given by a category $\mathbf{C}$, an endofunctor T and two natural transfomations η and μ:

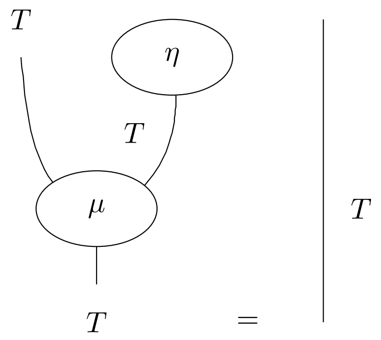


satisfying three equations, which we give here in graphical form:

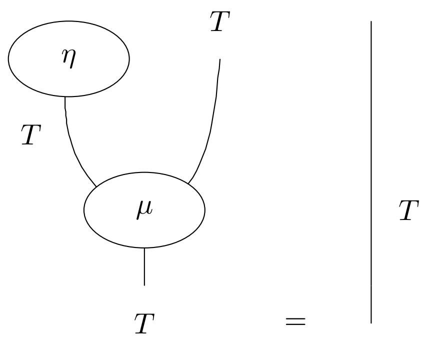
EQUATION $\mu - \mu$:



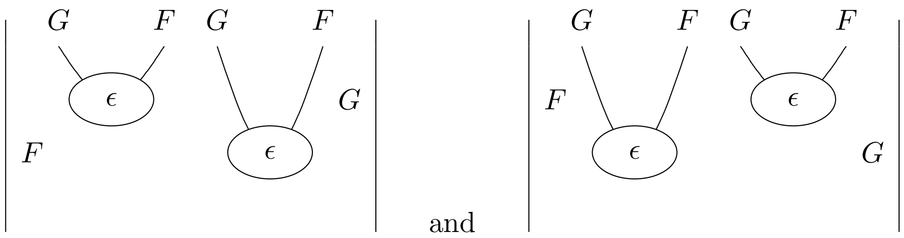
EQUATION $\mu - \eta$:



EQUATION $\eta - \mu$:

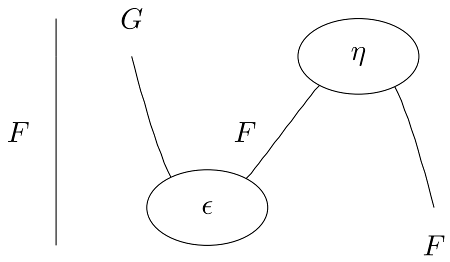


Let us check that the material we got above from an adjunction indeed satisfies the equation. The left and right members of equation $(\mu - \mu)$ are, respectively:



and hence are equal by Godement's rule.

The left hand side of $(\mu - \eta)$ is

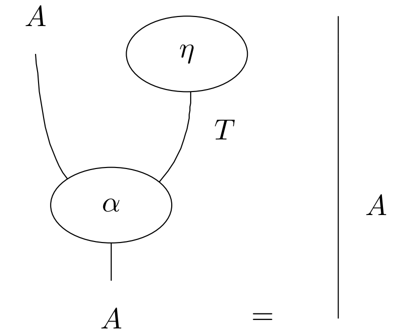


which reduces by $(\epsilon - \eta)$ to two wires F and G, that is, to a wire T, as required. The verification of $(\mu - \eta)$ is symmetric.

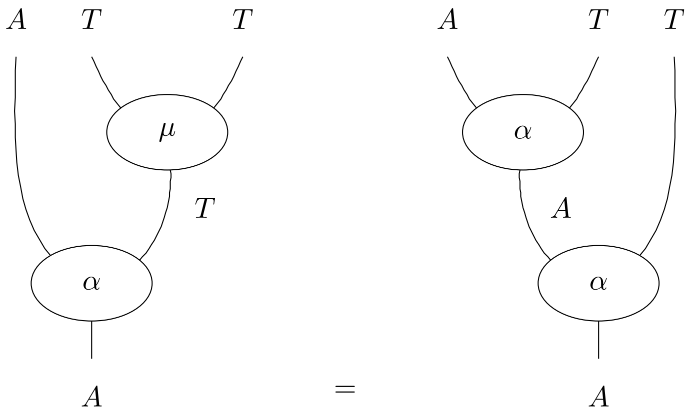
2.5.1 Algebras over a monad

Given a monad (T, η, μ) over a category $\mathbf{C}$, an algebra over (T, η, μ) is a pair (A, α) of an object of $\mathbf{C}$ and a morphism $\alpha : TA \to A$ that satisfies two axioms relating α with η and μ, respectively.

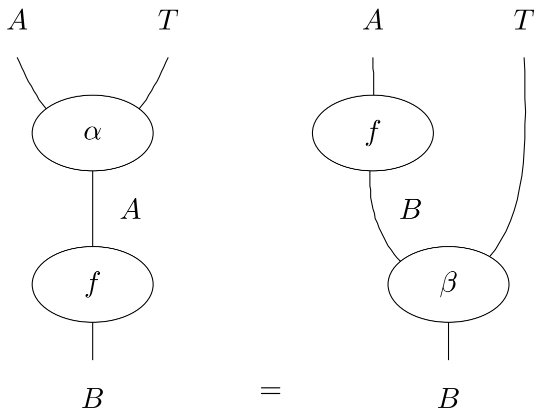
EQUATION $\alpha - \eta$:



EQUATION $\alpha - \mu$:



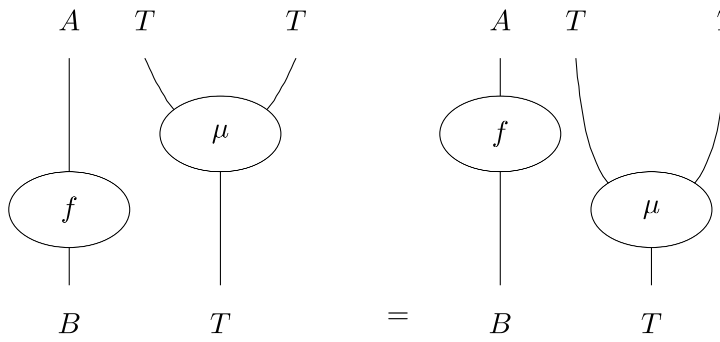
T-algebras are organised into a category $\mathbf{C}^T$, where $\mathbf{C}^T[(A,\alpha),(B,\beta)]$ is the set of morphisms $f \in \mathbf{C}[A,B]$ preserving the algebra structure, which amounts to the following equality:



We now construct an adjunction $F^T \dashv G^T$ between $\mathbf{C}$ and $\mathbf{C}^T$. We define:

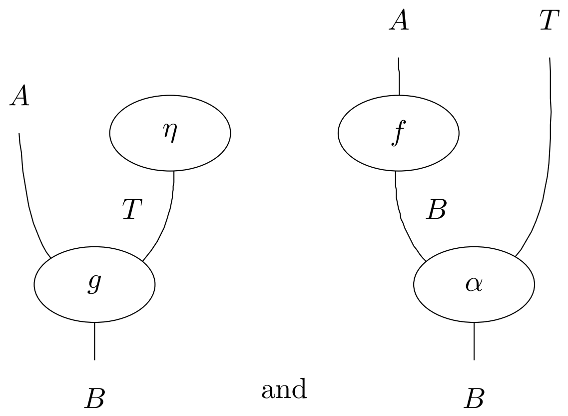
- $F^T(A) = (TA, \mu_A) \quad , \quad F^T(f) = Tf$,
- $G^T(A,\alpha) = A \quad , \quad G^T(f) = f$, i.e., G^T is the forgetful functor (in Chapter 4, we shall denote it with U, as usual).

It is clear that $GF = T$ and that G is a functor. We check that F^T is well-defined. The left and right hand sides of the equation $F^T(f) \circ \mu_A = \mu_B \circ T(F^T(f))$ are, respectively

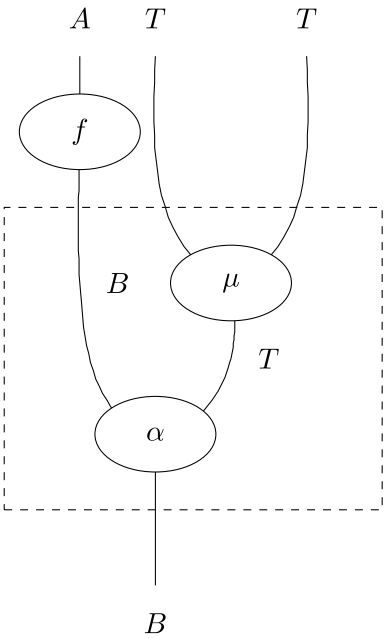


and we conclude using Godement's rule.

We next define $\zeta : \mathbf{C}^T[F^T A, (B, \alpha)] \to \mathbf{C}[A, B]$ and $\xi : \mathbf{C}[A, B] \to \mathbf{C}^T[F^T A, (B, \alpha)]$. Let $g : (TA, \mu_A) \to (B, \alpha)$ and $f : A \to B$. We define $\zeta(g)$ and $\xi(f)$ as follows:

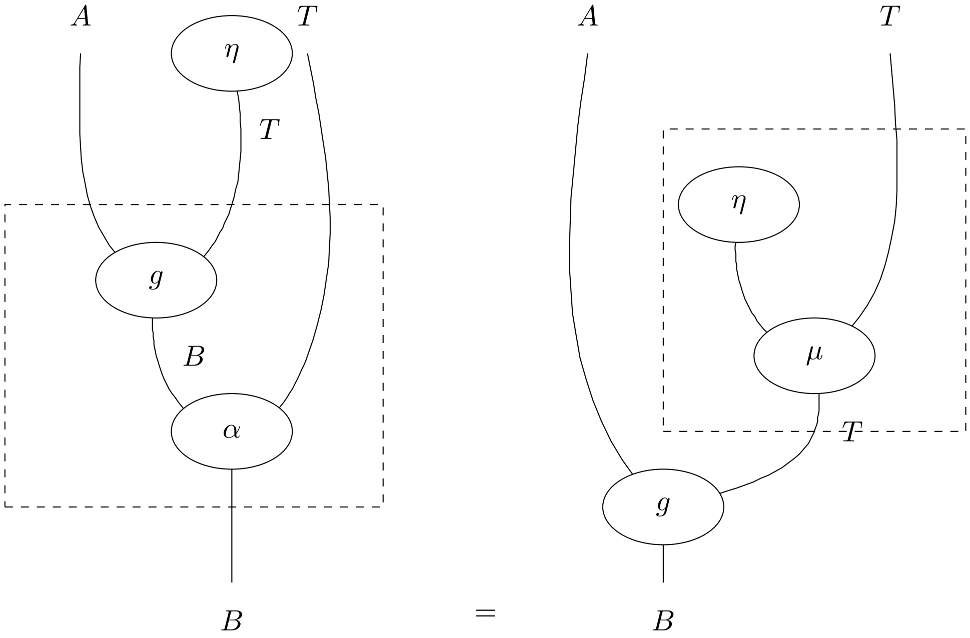


We show that $\xi(f)$ is indeed a morphism from (TA, μ_A) to (B, α), i.e. that $\xi(f) \circ \mu_A = \alpha \circ T(\xi(f))$. The diagram for the left hand side is



and by applying $(\alpha - \mu)$ as highlighted, we get $\alpha \circ T(\xi(f))$.

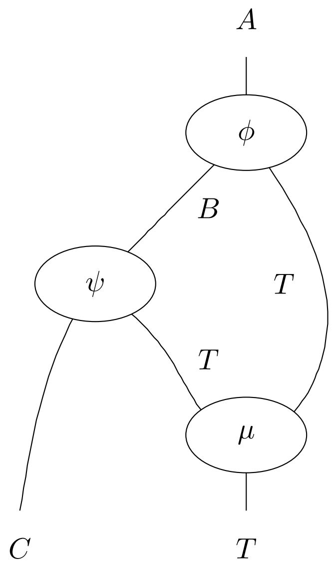
Next we check that ζ and ξ are inverse. Drawing $\zeta(\xi(f))$ and pushing f, the rest of the diagram shrinks to a wire B by applying $(\alpha - \eta)$, and thus $\zeta(\xi(f)) = f$. Conversely, we transform $\xi(\zeta(g))$ as follows (using the fact g is a T-algebra morphism):



We get g as required by transforming the $(\eta - \mu)$ redex.

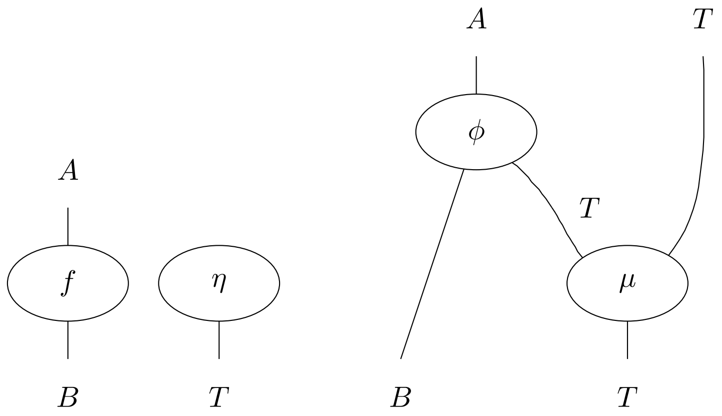
2.5.2 The Kleisli construction

We now provide another way to associate an adjunction with a monad. The *Kleisli category* $\mathbf{C}_T$ associated with a monad (T, η, μ) on a category $\mathbf{C}$ has the same objects as $\mathbf{C}$, while $\mathbf{C}_T[A, B] = \mathbf{C}[A, TB]$. The identity morphism at A is $\eta_A : A \to TA$, and the compostion of $\phi : A \to TB$ and $\psi : B \to TC$ is defined as follows:

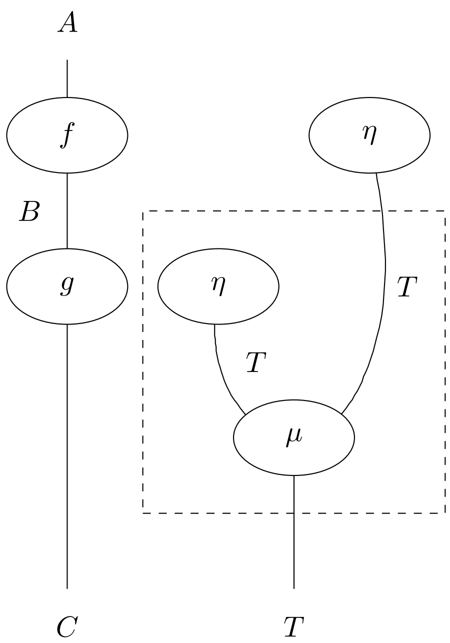


We construct an ajunction $F_T \dashv G_T$ between $\mathbf{C}$ and $\mathbf{C}_T$ as follows:

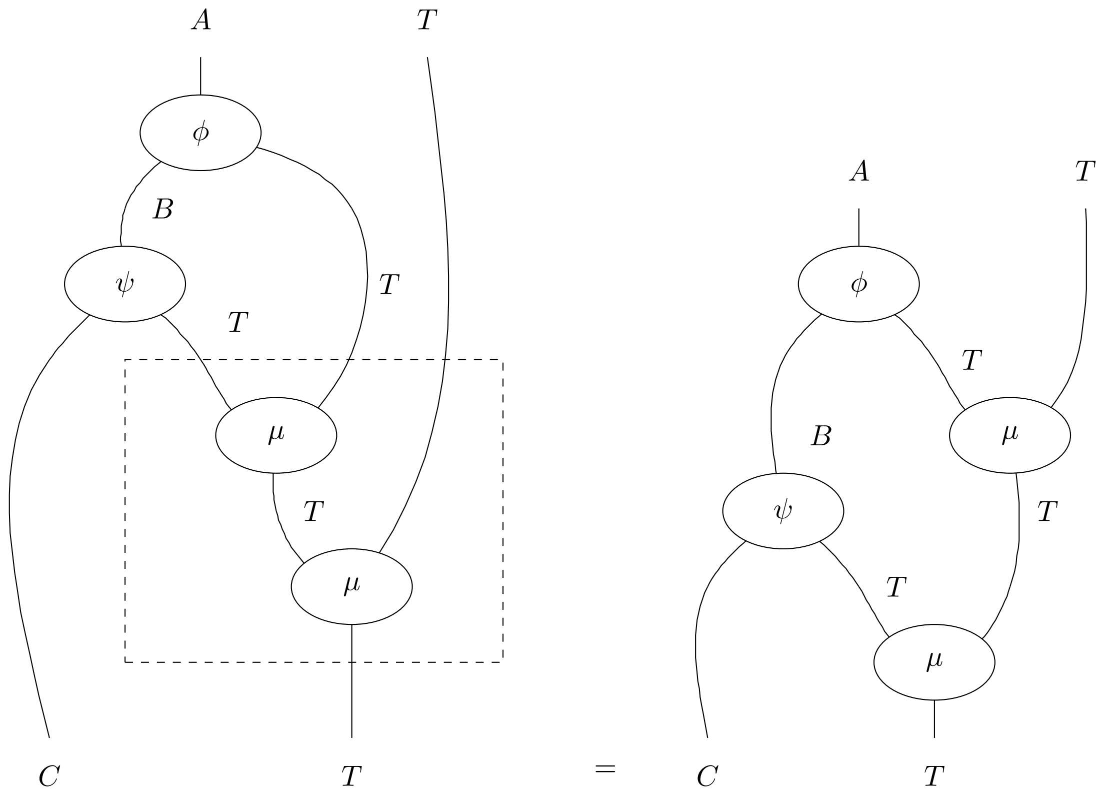
- $F_T(A) = A$ and $G_T(A) = TA$,
- $F_T(f)$ and $G_T(\phi)$ are as follows, respectively:



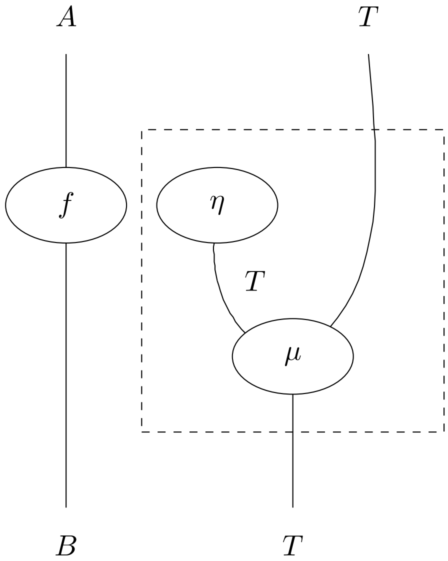
We check that F_T and G_T preserve the composition of morphisms. The composition of $F_T(f)$ and $F_T(g)$ in $\mathbf{C}_T$ is



from where we get $F_T(g \circ f)$ after applying $(\eta - \mu)$. As for G_T, the proof is as follows, using rule $(\mu - \mu)$:



We now check that $G_T F_T = T$. This is clear on objects. On morphisms, we get $G_T(F_T f) = Tf$ by applying $(\eta - \mu)$ as indicated below:

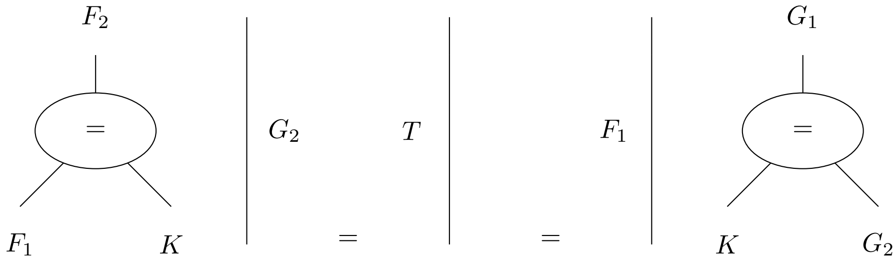


We have an ajunction situation by construction, since

$$\mathbf{C}_T[F_T A, B] = \mathbf{C}_T[A, B] = \mathbf{C}[A, TB] = \mathbf{C}[A, G_T B]$$

2.5.3 Initial and final adjunctions associated to a monad*

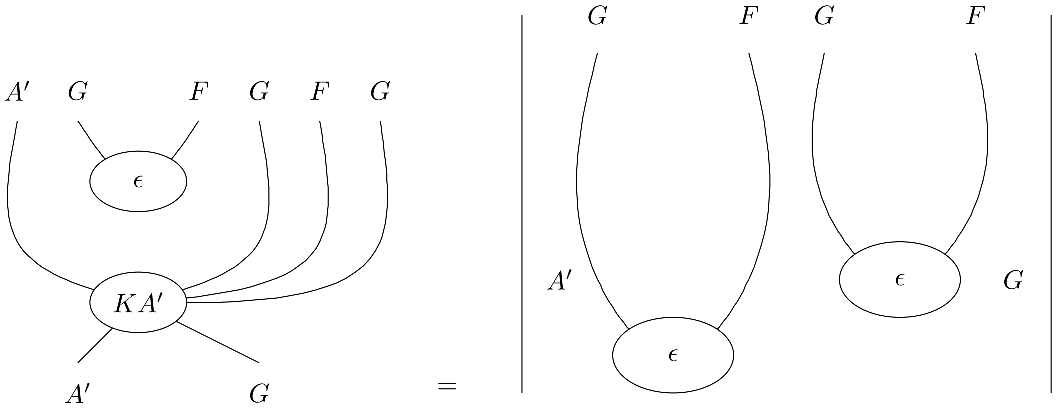
We show that the adjunctions constructed in the previous two subsections are "canonical" in the sense that the first one is final and the second one is initial in the category of all adjunctions that induce a given monad (T, η, μ) on a category $\mathbf{C}$. We first define this category as the subcategory of **Adj** (cf. subsection 2.2.1) whose objects are the adjunctions inducing T, and whose morphisms are of the form (id, K). Note that all adjunctions in this category are between $\mathbf{C}$ (fixed) and some $\mathbf{C}'$, and that they all have the same unit η. Note also that in the definition of morphism, the condition of commutation with units (and hence with counits) is satisfied for free. Indeed both hand sides of Equation (2.1) are equal to η, since



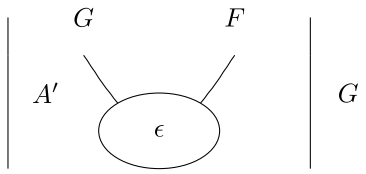
So, all we have to do now is to prove that, given any adjunction $F \dashv G$ between $\mathbf{C}$ and $\mathbf{C}'$ that induces (T, η, μ):

- there exists a unique functor $K^T : \mathbf{C}' \to \mathbf{C}^T$ such that $G^T K^T = G$ and $K^T F = F^T$, and
- there exists a unique functor $K_T : \mathbf{C}_T \to \mathbf{C}'$ such that $K_T F_T = F$ and $GK_T = G_T$,

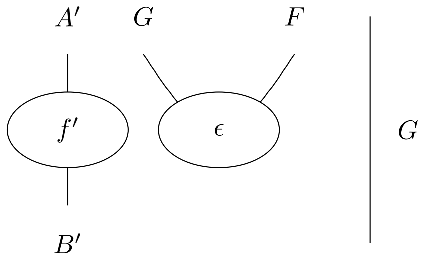
We first show the uniqueness of K^T. The equality $G^T K^T = G$ says that, for all A', $K^T A'$ has GA' as carrier and that K^T acts as G on arrows. The equality $K^T F = F^T$ imposes $K^T(FA) = (GFA, \mu_A)$ for all A, hence in particular $K^T(FGA') = (GFGA', \mu_{GA'})$. Further, $K^T(\epsilon_{A'}) = G\epsilon_{A'}$ has to be a morphism from $K(FGA')$ to KA', i.e., the following string diagrams have to be equal (remember that $\mu_A = G(\epsilon_{FA})$, where ϵ is the counit of the adjunction $F \dashv G$):



Now we can plug $\eta_{GA'}$ on the top on both sides, which leaves us after reduction with a (uniquely determined) definition of $K^T A'$:



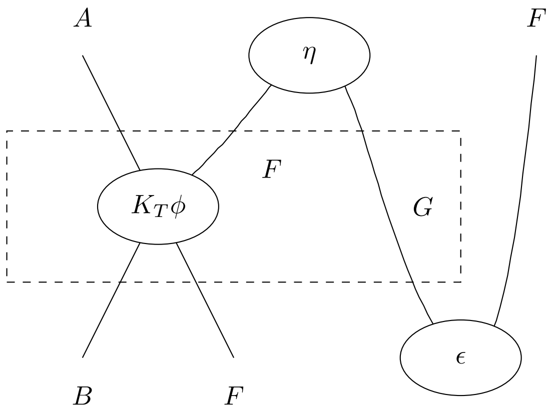
It remains to show that K^T "exists", i.e., is well-defined and satisfies the required properties. We check that, for any $f' : A' \to B'$, $K^T f'$ is a morphism from $G\epsilon_{A'}$ to $G\epsilon_{B'}$. Indeed, both $K^T f' \circ G\epsilon_{A'}$ and $G\epsilon_{B'} \circ T(K^T f')$ read as



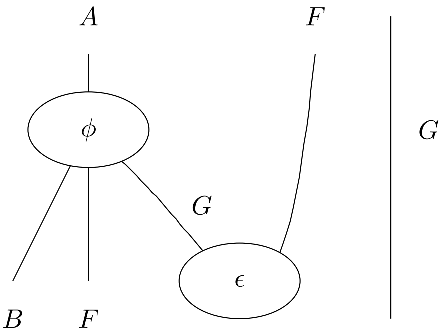
We have $G^T K^T = G$ by construction, and $K^T F = F^T$ is also easy:

- $K^T FA = G\epsilon_{FA} = \mu_A$ (since $F \dashv G$ induces T);
- $K^T Ff = GFf = Tf = F^T f$.

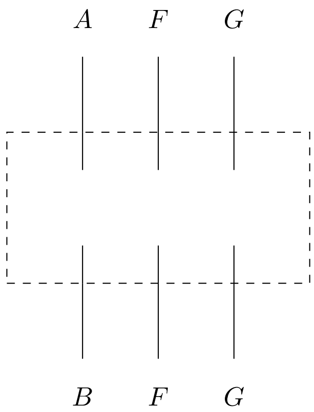
We now synthesize K_T. The equality $K_T F_T = F$ imposes $K_T A = FA$. The equality $GK_T = G_T$ imposes $GK_T\phi = G_T(\phi)$. Now, by $(\eta - \epsilon)$ expansion, $K_T\phi$ can be represented as



By this trick we can use our contextual information about $K_T(\phi)$, namely that $GK_T\phi = G_T(\phi)$, and replace the highlighted region with

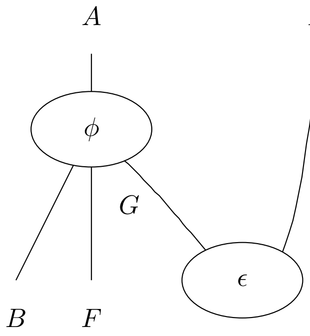


We pause here a second to note that both diagrams – the previous one, and the region above – indeed have the same interface, namely

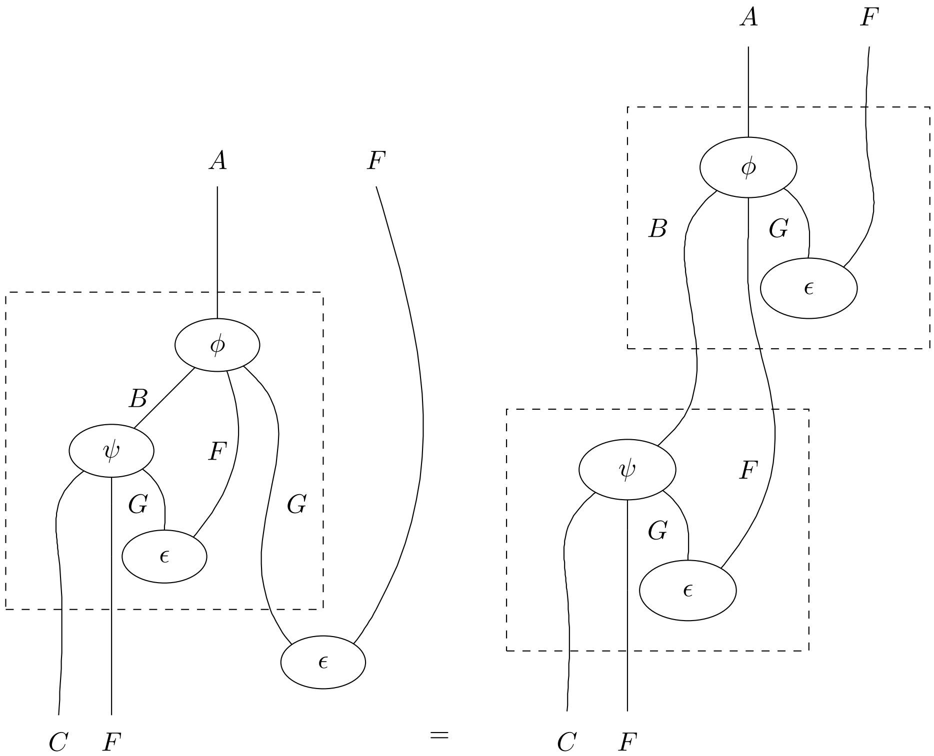


It is a good discipline to make this kind of type-checking often when reasoning with string diagrams.

Coming back to our synthesis, we notice that after replacement it is still possible to undo the $(\eta - \epsilon)$ expansion, since the former and the new region have the same shape on the right: just two wires G and F. And this leads us to the following (uniquely determined) value of $K_T\phi$:

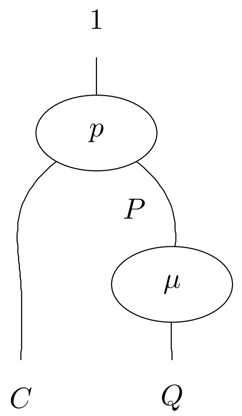


The preservation of composition is a simple game of naturality, as illustrated below:



2.6 Presheaves and string diagrams

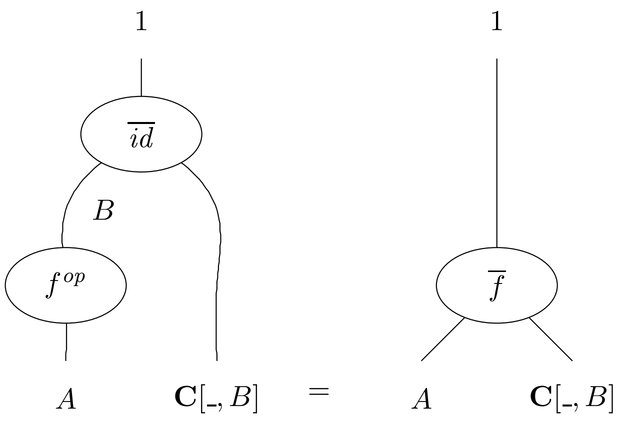
In this section, we revisit Yoneda lemma and the result that every presheaf is a colimit of representable presheaves, using the language of string diagrams. Let $P, Q : \mathbf{C}^{op} \to \mathbf{Set}$ be presheaves, and consider a natural transformation $\mu : P \to Q$. We would like to be able to draw diagrams not only for μ_C, but also for $\mu_C(p)$ for some $p \in PC$. This is possible if we consider p as a morphism from 1 (the terminal object of **Set**) to PC:



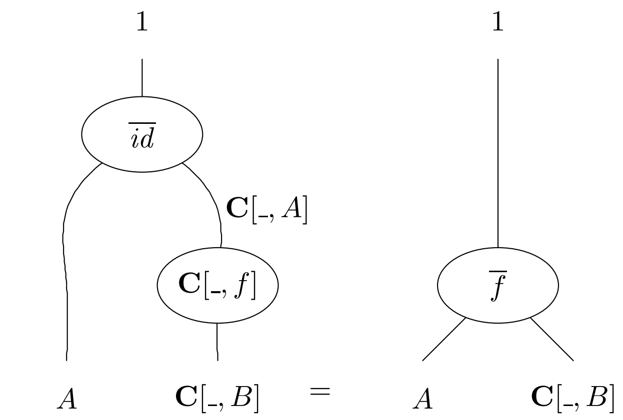
This diagram involves three categories: the terminal category **1** on the left, the category $\mathbf{C}^{op}$ in the middle, and the category **Set** on the right.

When $P = \mathbf{C}[\_, B]$, then p is a morphism from C to B. But since it is here viewed as a morphism from 1 to $\mathbf{C}[\_, B]C$, we shall use an explicit coercion and signal it with an overlining. We next give a diagrammatic definition of the left and right homfunctors, respectively.

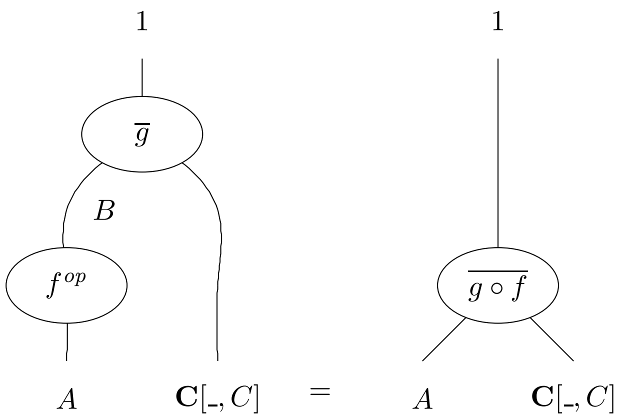
EQUATION *Homleft*:



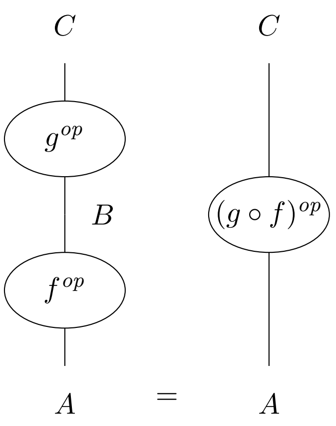
EQUATION *Homright*:



These two equations define the action of $\mathbf{C}[f, B]$ and of $\mathbf{C}[A, f]$ on the identity morphism, but this is enough to derive the full definition of both homfunctors. We get

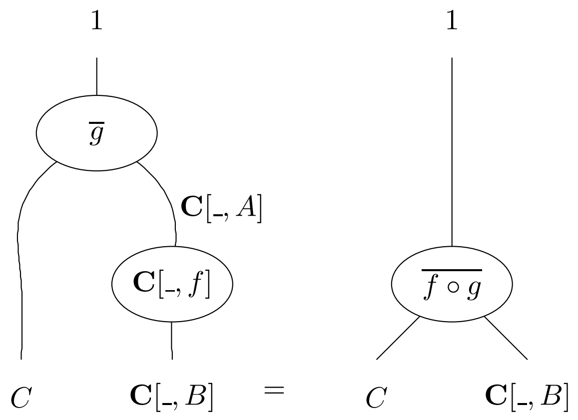


by applying Equation (*Homleft*) on both sides, and then the equality

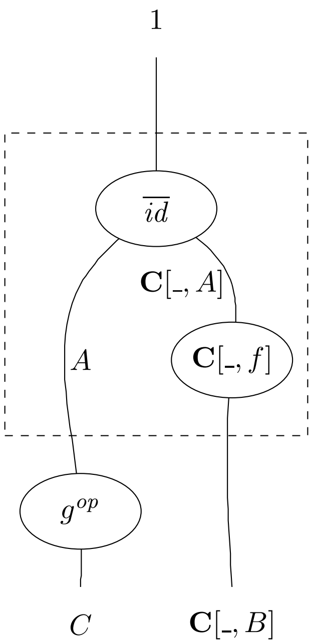


In our description of the left homfunctors, the reversal of f, g witnesses that these functors are contravariant.

The full definition of the right homfunctor is the following:

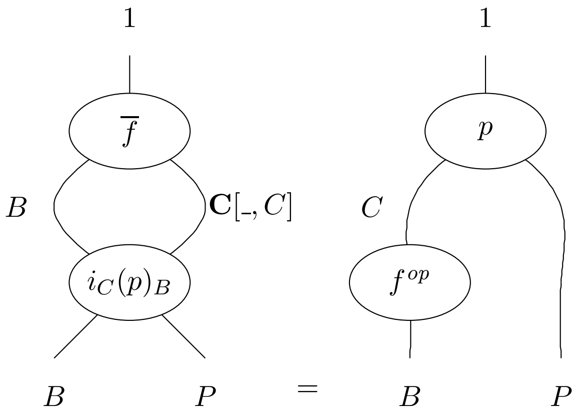


and we derive it by first transforming the left hand side thanks to equation *Homleft*, as follows:

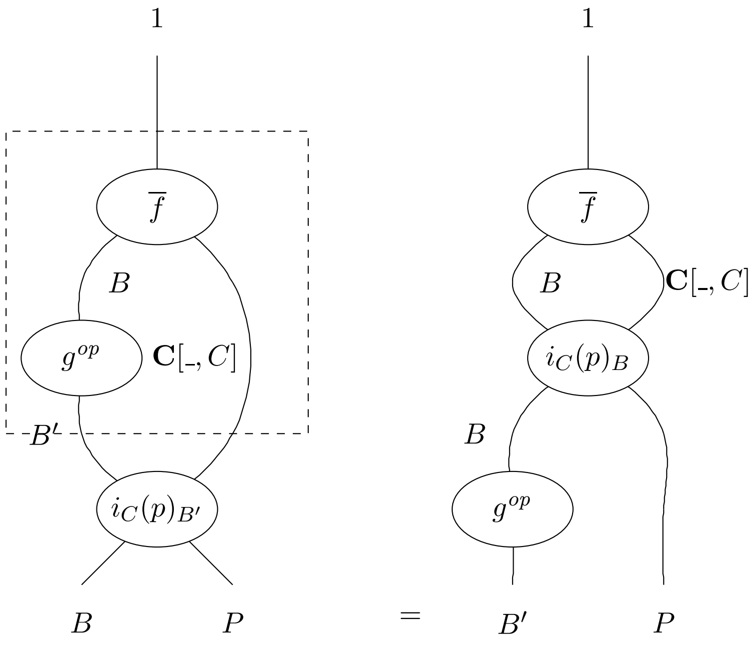


and then applying Equation (*Homright*) to the highlighted region, which leaves us with the equation defining the full action of the left homfunctor, that we have already established.

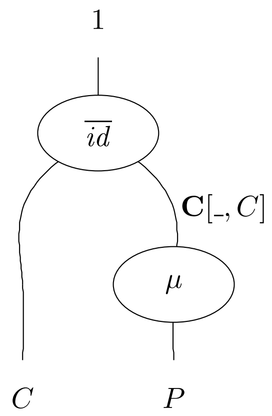
We now give a graphical proof of Yoneda lemma. Let $P : \mathbf{C}^{op} \to \mathbf{Set}$ and $p \in PC$. We define $i_C(p) : \mathbf{Set}^{\mathbf{C}^{op}}[\mathbf{C}[\_, C], P]$ as follows:



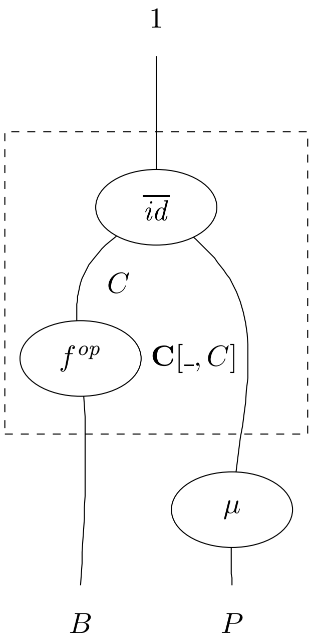
The naturality of $i_C(p)$ is expressed as:



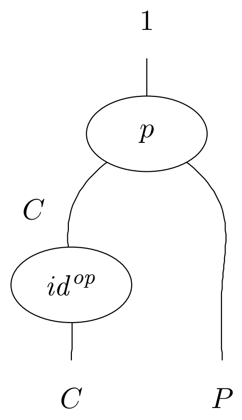
in which the highlighted region is equal to $f \circ g$ by the definition of $C[g, C]$ (cf. above), from which the naturality equality easily follows, by unfolding the definition of $i_C(p)$. Now let $\mu : \mathbf{Set}^{\mathbf{C}^{op}}[\mathbf{C}[\_, C], P]$. We define $j_C(\mu)$ as



We check that i_C and j_C are inverse bijections. Pushing f^{op} up and μ down, $i_C(j_C(\mu))_B(f)$ is



which by *Homleft* is $\mu_B(f)$. Conversely, $j_C(i_C(p))$ is



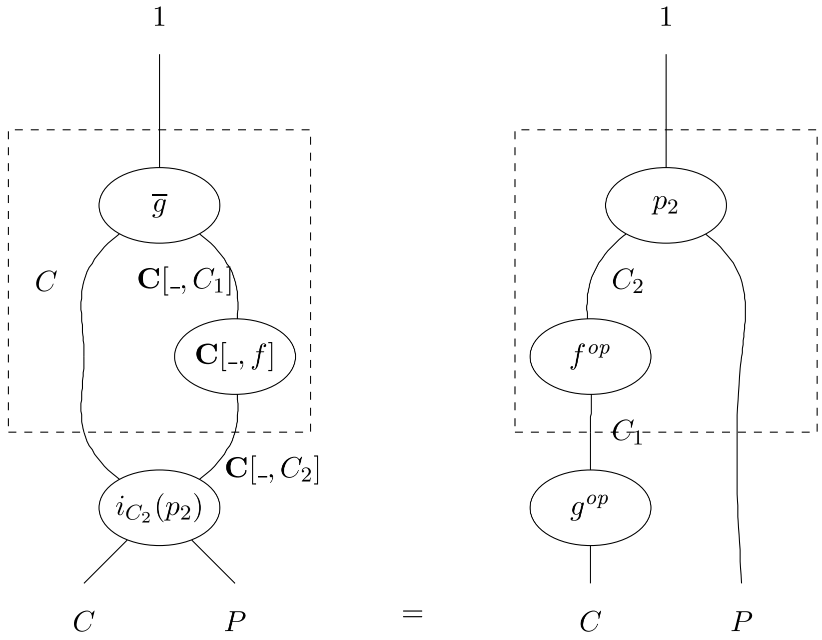
which is p.

We prove now graphically that every presheaf is a colimit of representable presheaves.

Proof of Proposition 1.4.3. Let $P : \mathbf{C}^{op}$. We consider the diagram formed by the $\mathbf{C}[\_, C]$'s indexed over all (C, p) where $p \in PC$. In other words, we have a copy of $C[\_, C]$ for each p. The morphisms of the diagram are all the $\mathbf{C}[\_, f]$'s between a copy of $\mathbf{C}[\_, C_1]$ at p_1 and a copy of $\mathbf{C}[\_, C_2]$ at p_2 such that $p_1 = Pfp_2$. We show that the family of the $i_C(p)$'s provides us with a cocone to P. First, for each (C, p), we have indeed $i_C(p) : \mathbf{C}[\_, C] \to P$. We have to show that, for all C_1, C_2, f, p_2, the following triangle commutes:

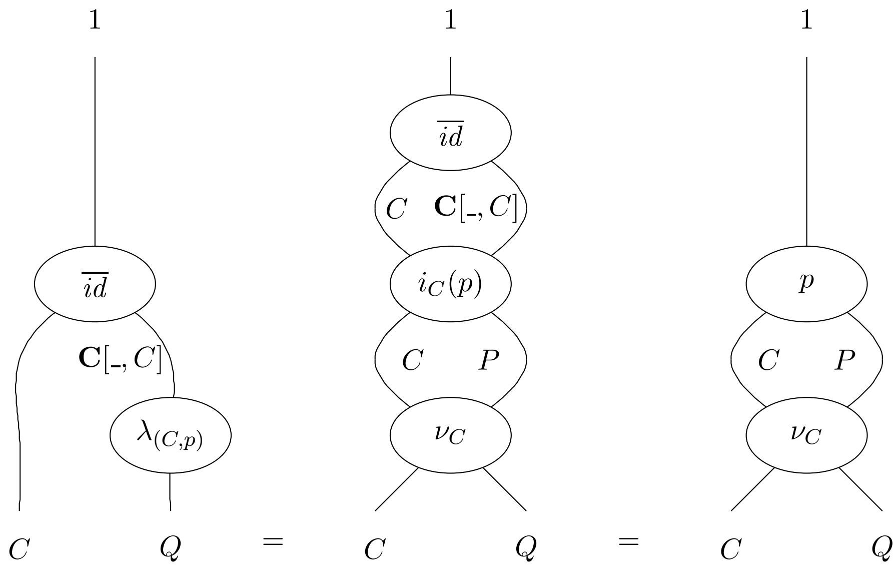
$$\begin{tikzcd} & P & \\ \mathbf{C}[\_, C_1] \arrow[ur, "i_{C_1}(Pfp_2)"] \arrow[rr, "\mathbf{C}[\_, f]"] & & \mathbf{C}[\_, C_2] \arrow[ul, "i_{C_2}(p_2)"'] \end{tikzcd}$$

i.e., that, for all C, g (and turning to string diagrams):



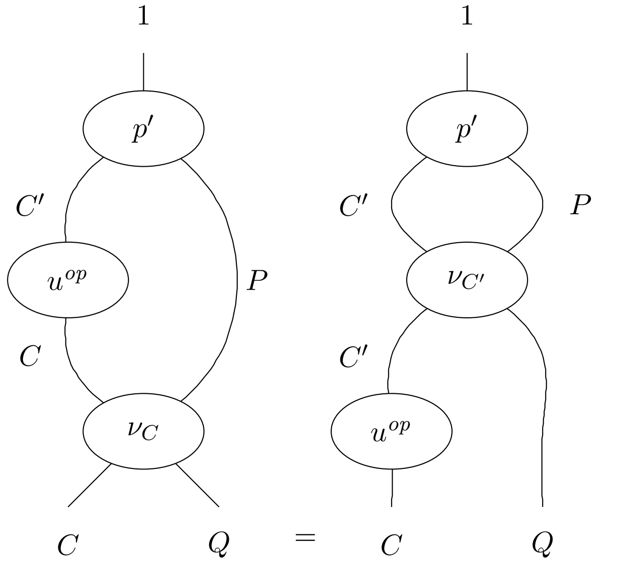
We have highlighted a region on the right to ease the parsing of $(i_{C_1}(Pfp_2))_C(g)$. The region on the left can be transformed to $f \circ g$, and the conclusion follows by definition of $i_{C_2}(p_2)$ since $(f \circ q)^{op} = g^{op} \circ f^{op}$.

Consider now an arbitrary cone formed by a family of $\lambda_{(C,p)}$'s with codomain Q. We look for ν such that $\nu \circ i_C(p) = \lambda_{(C,p)}$ for all (C, p). In particular, we have $(\lambda_{(C,p)})_C(id) = \nu_C((i_C(p))_C(id)$:



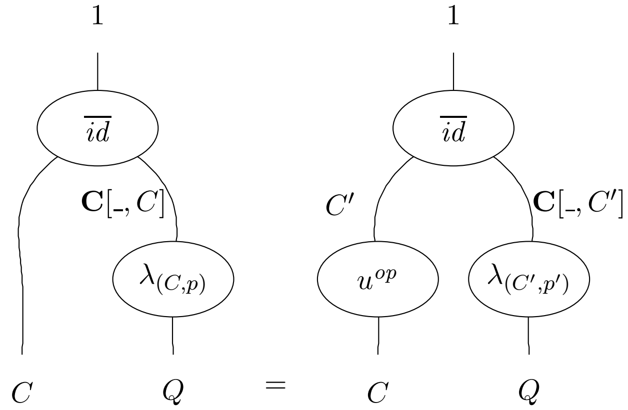
where the right hand side provides us with the (uniquely determined) definition of ν.

We check that ν is natural, namely that we have, for all u, p':

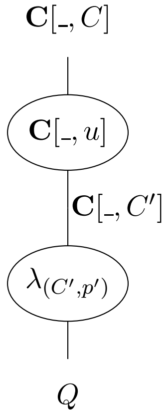


(2.12)

We set $Pup' = p$ and we apply the definition of ν, so that the equality to be proved is:



Since λ is a cocone, we can replace $\lambda_{(C,p)}$ with

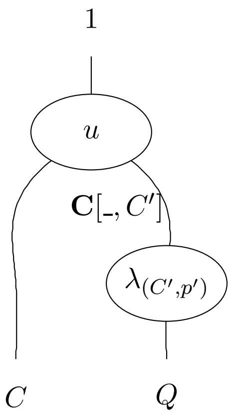


and we conclude using Equations (*Homleft*) and (*Homright*).

We are left to check that ν is a cocone morphism, i.e., that the following triangle commutes, for all (C', p'):

$$\begin{tikzcd}
& Q & \\
& & P \arrow[ul, "\nu"'] \\
\mathbf{C}[_, C'] \arrow[uu, "\lambda_{(C',p')}"] \arrow[urr, "i_{C'}(p')"'] & &
\end{tikzcd}$$

The drawing for $\nu \circ i_{C'}(p')$ at C, u is the left hand side of Equality (2.12), which we have to match with



which can be transformed into the right hand side of Equality (2.12) by applying Equation (*Homleft*) to u. This completes the proof. $\square$

Chapter 3

Kan extensions

In this chapter, we introduce left and right Kan extensions, that allow us to capture in a unified framework the notions of limits, adjunctions, as well as the Yoneda embedding. In Section 3.1, we define Kan extensions and we show how to construct them in terms of limits and colimits. In Section 3.2, we introduce dinatural transformations, ends and coends, which play a role similar to natural transformations, limits and colimits for contravariant/covariant functors and we recast the constructions of Section 3.1 in these terms. In Section 3.3, we discuss special cases.

3.1 Kan extensions: definitions and constructions

Recall that in the definition of adjunction, knowing, say, a functor $G : \mathbf{C}' \to \mathbf{C}$, it is useful to give a name to the pair $(FA, \eta_A : A \to GFA)$: it is called the universal pair from A to G (cf. Definition 1.5.4). Likewise, when defining a right adjoint to a functor $F : \mathbf{C} \to \mathbf{C}'$, the pair $(GA', \epsilon_{A'} : FGA' \to A')$ is called couniversal from F to A'. This terminology becomes useful when the universal pair does not exist for every A. Then G may have no left adjoint, but still a universal pair may exist for certain objects A. Or it may just be that the focus of the discussion concerns a particular object A.

Let $K : \mathbf{M} \to \mathbf{C}$ be a functor, and let $\mathbf{A}$ be a category. We consider the following problem: does the functor $\mathbf{A}^K : \mathbf{A}^{\mathbf{C}} \to \mathbf{A}^{\mathbf{M}}$ have left and right adjoints? The corresponding universal and couniversal problems are the following.

- A right Kan extension of $T : \mathbf{M} \to \mathbf{A}$ along K is given by a functor $Ran_K(T) : \mathbf{C} \to \mathbf{A}$ and a natural transformation $\epsilon_T : Ran_K(T)K \to T$ such that $(Ran_K(T), \epsilon_T)$ is couniversal from $\mathbf{A}^K$ to T.

- A left Kan extension of $T : \mathbf{M} \to \mathbf{A}$ along K is given by a functor $Lan_K(T) : \mathbf{C} \to \mathbf{A}$ and a natural transformation $\eta_T : T \to Lan_K(T)K$ such that $(Lan_K(T), \eta_T)$ is universal from T to $\mathbf{A}^K$.

The following diagrams illustrate the situation:

$$
\begin{tikzcd}
\mathbf{M} \arrow[r, "K"] \arrow[d, "T"'] & \mathbf{C} \arrow[dl, "Ran_K(T)"] \arrow[dl, phantom, "\Leftarrow \epsilon", bend right] \\
\mathbf{A} &
\end{tikzcd}
\qquad
\begin{tikzcd}
\mathbf{M} \arrow[r, "K"] \arrow[d, "T"'] & \mathbf{C} \arrow[dl, "Lan_K(T)"] \arrow[dl, phantom, "\Rightarrow \eta", bend right] \\
\mathbf{A} &
\end{tikzcd}
$$

We will show that provided $\mathbf{A}$ has enough limits, the right Kan extension exists, and is defined by the following formula:

$$Ran_K TC = \lim(T\pi) \qquad \epsilon_M = \lambda_{id_{KM}}$$

where $\pi : (C \downarrow K) \to \mathbf{M}$ is the projection functor, where $C \downarrow K$ is the category whose objects are the pairs $(M, f : C \to KM)$, with

$$(C \downarrow K)[(M, f), (M', g)] = \{h : M \to M' \mid g = (Kh)f\}$$

and where $(\lim(T\pi), \lambda)$ is the limit cone. In order to understand the definition of ϵ, we note that for $C = KM$ the diagram $T\pi$ is indexed by the morphisms from KM to KM', among which is id_{KM}. The idea underlying the indexation is that of approaching C by objects of M, so as to be able to "extend" T from $\mathbf{M}$ to $\mathbf{C}$. This is only an image, as we do not seek to have $T = (Ran_K T)K$, in general (but see below for an important special case).

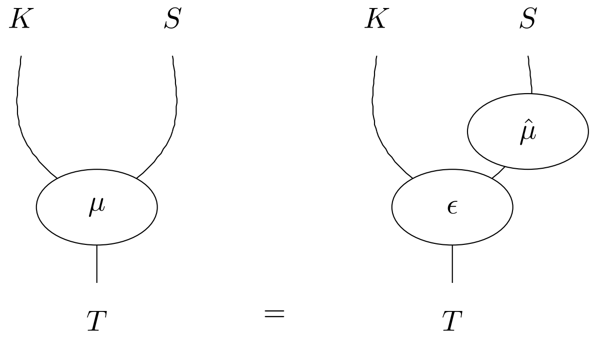
We first observe that the universal property of a limit allows us to define the action of $Ran_K T$ on morphisms. Let $g : C \to C'$. If λ and λ' are the cones respectively associated with the limits $Ran_K TC$ and $Ran_K TC'$, we build a new cone λ'' by composition with g, i.e., $\lambda''_f = \lambda_{fg}$, and we define $Ran_K Tg$ as the mediating arrow from λ'' to λ', so that we have

$$\lambda'_f \circ Ran_K Tg = \lambda_{fg} \text{ for all } f : C' \to KM \tag{3.1}$$

In particular, for $C' = KM$ and $f = id$, we obtain

$$\epsilon_M \circ Ran_K Tg = \lambda_g \tag{3.2}$$

Let $\mu : SK \to T$. We seek a morphism $\hat{\mu} : S \to Ran_K T$ such that $\epsilon \circ \hat{\mu} K = \mu$:



In order to synthesize $\hat{\mu}_C$, it is enough to find a cone from SC to the TM's, indexed by the morphisms $f : C \to KM$. We take $\mu_M \circ Sf$ as component at f. This indeed defines a cone, i.e., we have $Th \circ (\mu_M \circ (Sf)) = \mu_{M'} \circ S((Kh)f)$ for all $h : M \to M'$, by naturality of μ. We then define $\hat{\mu}_C$ as the mediating arrow associated with this cone. We have by construction $\lambda_f \circ \hat{\mu}_C = \mu_M \circ Sf$. In particular, for $C = KM$ and $f = id$, we have $\epsilon_M \circ \hat{\mu}_{KM} = \mu_M$, i.e., $\epsilon \circ \hat{\mu}K = \mu$. We are left to show that $\hat{\mu}$ is unique. Let $\mu' : S \to Ran_K T$ be such that $\epsilon \circ \mu' K = \mu$. It is enough to prove that μ'_C satisfies the characteristic property of $\hat{\mu}_C$, that is, $\lambda_f \circ \mu'_C = \mu_M \circ Sf$, or equivalently, using Equation 3.2:

$$\epsilon_M \circ Ran_K Tf \circ \mu'_C = \mu_M \circ Sf$$

This follows from the naturality of μ' and from the assumption on μ'.

Note that we did not need any property of **C**, we only had to assume the existence of some limits in **A**.

We show now that when K is full and faithful, ϵ is iso, and hence Ran_K is full and faithful. For all $g : KM \to KM'$, we have a unique $g' : M \to M'$ such that $g = Kg'$. We then build a cone of vertex TM by taking Tg' as component at g. We use the faithfulness of K to prove that this is indeed a cone. We now show that the cone is also limiting. It is easy to check that the property that K is full and faithful is equivalent to the property that (M, id) is initial in $KM \downarrow K$, for all M. Then the claim follows from the following general result.

Proposition 3.1.1 *If $F : \mathbf{D} \to \mathbf{C}$ is a diagram and if $\mathbf{D}$ has an initial object I, then FI is a limit of F, together with the cone formed by the family of the morphisms $F(f)$ where the f's are the unique morphisms of source I to the objects of $\mathbf{D}$.*

PROOF. Left to the reader. □

We next note that by construction the limiting cone of vertex TM has the identity as component at id, and since TM is $Ran_K T(KM)$ up to iso (all limits of a given diagram are isomorphic), we have proved that ϵ is iso. We can even choose (as we just did) $Ran_K T$ in such a way that $Ran_K T \circ K = T$ and $\epsilon = id$.

Left Kan extensions are obtained similarly, replacing limits by colimits:

$$Lan_K TC = \text{colim } (T\pi) \qquad \eta_M = \lambda_{id_{KM}}$$

where now π the projection functor from the category $K \downarrow C$ whose objects are the pairs $(M, f : KM \to C)$, and where $(\text{colim } (T\pi), \lambda)$ is the colimit cone.

If K is full and faithful, we also have that η is iso, and hence that Lan_K is full and faithful.

3.2 Dinatural transformations

We pause here to define a notion similar to that of natural transformation, which we shall put to use to give alternative, handier formulas for manipulating Kan extensions.

There are many families of morphisms indexed by the objects of a category that do not form a natural transformation becase of a conflit between covariance and contravariance. For example, in a CCC, we would like to say that $eval : B^A \times A \to B$ is "natural" in A. But A appears twice in the type of $eval$, once in a contravariant position and once in a covariant position. We shall develop several notions, summarized by the following correspondence table:

dinatural transformation	natural transformation
wedge	cone
end, coend	limit, colimit

Let $S, T : \mathbf{C}^{op} \times \mathbf{C} \to \mathbf{C}'$. A dinatural transformation is a family of morphisms α_C indexed by the objects of $\mathbf{C}$ such that for all $C, C', f : C \to C'$ we have:

$$T(id, f) \circ \alpha_C \circ S(f, id) = T(f, id) \circ \alpha_{C'} \circ S(id, f)$$

This commuting diagram is called Mac Lane's hexagon:

$$\begin{tikzcd}
 & S(C,C) \arrow[r, "\alpha_C"] & T(C,C) \arrow[dr, "T(C,f)"] & \\
S(C',C) \arrow[ur, "S(f,C)"] \arrow[dr, "S(C',f)"'] & & & T(C,C') \\
 & S(C',C') \arrow[r, "\alpha_{C'}"'] & T(C',C') \arrow[ur, "T(f,C')"'] &
\end{tikzcd}$$

Natural transformations can be seen as a special case of dinatural transformations: natural transformations between two functors $F, G : \mathbf{C}^{op} \to \mathbf{C}'$ are in one-to-one correspondence with the dinatural transformations between $\lambda(x, y).Fx$ and $\lambda(x, y).Gx$.

But there is a serious limitation of this parallel development: dinatural transformations do not compose vertically (try!). However, the composition of a natural transformation and a dinatural transformation (or in the reverse order) is easily seen to be a dinatural transformation, i.e., say, given $\mu : F \to G$ and $\nu : \lambda(x, y).Gx \to S$, then $\nu \circ \mu$ (defined pointwise) is a dinatural transformation from $\lambda(x, y).Fx$ to S. These compositions will be enough for our purposes.

Next, we replace diagrams $F : \mathbf{D} \to \mathbf{C}$ by functors $F : \mathbf{D}^{op} \times \mathbf{D} \to \mathbf{C}$. The constant functors ΔC are now defined from : $\mathbf{D}^{op} \times \mathbf{D}$ to $\mathbf{C}$. A wedge from C to F is a dinatural transformation from ΔC to F. An end of F is a pair $(C, \lambda : \Delta C \to F)$ that is couniversal from Δ to F. Note that the compositions involved in this definition are well-defined, since we can write ΔC as $\lambda(x,y).\Delta Cx$. We use the notation $\int_d F(d,d)$ for C.

A coend is defined similarly as a universal wedge from F to Δ, whose vertex is denoted by $\int^d F(d,d)$.

As an illustration, we prove the following formula:

$$(\mathbf{C}'^{\mathbf{C}})[U,V] = \int_c \mathbf{C}'[Uc, Vc] \tag{3.3}$$

for arbitrary $U,V : \mathbf{C} \to \mathbf{C}'$. A wedge from X to $\mathbf{C}'[U\_, V\_]$ is a family of fonctions $\lambda_C : X \to \mathbf{C}'[UC, VC]$ such that

$$\mathbf{C}'[UC_1, Vf] \circ \lambda_{C_1} = \mathbf{C}'[Uf, VC_2] \circ \lambda_{C_2}$$

for all $C_1, C_2, f : C_1 \to C_2$. This amounts to say that for all $x \in X$ the family $\lambda_C(x)$ indexed by the objects of $\mathbf{C}$ defines a natural transformation from F to G: we have thus defined a fonction from X dans $(\mathbf{C}'^{\mathbf{C}})[U,V]$. The limiting wedge is defined by taking $\lambda\mu.\mu_C$ as component at C.

Going back to Kan extensios, by grouping the copies of objects TM, we obtain the following equivalent formulations for $Ran_K T$ and $Lan_K T$ in terms of ends and of coends:

$$Ran_K TC = \int_M TM^{\mathbf{C}[C,KM]} \qquad Lan_K TC = \int^M \mathbf{C}[KM, C] \cdot Tm$$

where $TM^{\mathbf{C}[C,KM]}$ (resp. $\mathbf{C}[KM,C]\cdot TM$) denotes the product (resp. the coproduct) of as many copies of TM as there are morphisms in $\mathbf{C}[C, KM]$ (resp. $\mathbf{C}[KM, C]$).

3.3 Kan extensions are everywhere

We retrieve a number of fundamental notions as instantiations of that of Kan extension.

3.3.1 Limits

If $\mathbf{C}$ is the terminal category, then $C \downarrow K$ reduces to $\mathbf{M}$, and the right (respectively left) Kan extension boils dow to the limit (respectively colimit) of T.

3.3.2 Adjunctions

If $\mathbf{M} = \mathbf{A}$ and $T = id$, it is tempting to see there the notion of adjunction. But the situation is a bit more complex. In order to make the link precise, we need a definition.

Definition 3.3.1 *Let $F : \mathbf{A} \to \mathbf{A}'$. We say that F preserves the right Kan extension $(Ran_K T, \epsilon)$ if $(F Ran_K T, F\epsilon)$ is a right Kan extension of FT along K.*

Proposition 3.3.2 *The following properties are equivalent, for $K : \mathbf{A} \to \mathbf{C}$ and $R : \mathbf{C} \to \mathbf{A}$:*

1. *$R \dashv K$, with ϵ as counit.*

2. *(R, ϵ) is a right Kan extension of id along K that is preserved by all functors.*

3. *(R, ϵ) is a right Kan extension of id along K that is preserved by K.*

PROOF. (1) $\Rightarrow$ (2). Let $\mu : SK \to id$. We look for $\mu' : S \to R$ such that $\epsilon \circ \mu' K = \mu$. Writing this equality as a string diagram, and plugging the unit η, we see that μ' is necessarily equal to $\mu R \circ S\eta$ and that this definition of μ' fits. Let now $F : \mathbf{A} \to \mathbf{A}'$, and let $\mu : HK \to F$. We see likewise that $\mu R \circ H\eta$ is the unique transformation from H to FR that fits.

(3) $\Rightarrow$ (1). We apply the assumption that $(KR, K\epsilon)$ is a Kan extension, with $id : \mathbf{C} \to \mathbf{C}$. This yields a transformation η that satisfies one of the two laws of adjunction. For the second one, we observe that $\nu = \epsilon R \circ R\eta$ satisfies the condition $\epsilon \circ \nu K = \epsilon$ and hence $\nu = id$ by uniqueness.

3.3.3 Yoneda lemma

If $\mathbf{M} = \mathbf{C}$ and $K = id$, it is easy to see that (T, id) is both a left and right Kan extension of T along id, using the fact that K is a fortiori full and faithful. For what concerns the right extension, we have thus

$$TC = \int_{C'} (TC')^{\mathbf{C}[C,C']}$$

and when $\mathbf{A} = \mathbf{Set}$, then $(TC')^{\mathbf{C}[C,C']}$ is the set of all fonctions from $\mathbf{C}[C, C']$ to TC', I.e., we have $(TC')^{\mathbf{C}[C,C']} = \mathbf{Set}[\mathbf{C}[\_, C'], TC']$. Therefore, by 3.3, $\int_{C'} TC'^{\mathbf{C}[C,C']}$ is the set of natural transformations from $\mathbf{C}[C, \_]$ to T. We have thus retrieved Yoneda lemma as instance of the right Kan extension construction.

3.3.4 Dense functors

In this section, we consider two further sepcial cases of Kan extensions (Definition 3.3.3 and Proposition 3.3.4).

Definition 3.3.3 *If $\mathbf{A} = \mathbf{C}$, $T = K$ and $(id : \mathbf{C} \to \mathbf{C}, id : K \to K)$ is a left Kan extension of K along K, we say that K is dense in $\mathbf{C}$.*

Indeed, this definition amounts to say that for any object C of $\mathbf{C}$, the evident cone from $K\pi$ to C (whose component at f is f) is a colimit (using a "left" version of Equation 3.1).

Proposition 3.3.4 *Let* $\mathbf{C} = \mathbf{Set}^{\mathbf{M}^{op}}$ *and* $K = Y$ *(the Yoneda embedding). Then* $Lan_Y T$ *exists if and only if the curried version of* $\mathbf{A}[T\_,\_] : \mathbf{A} \times \mathbf{M}^{op} \to \mathbf{Set}$, *i.e., the functor* R_T *defined by* $R_T(A) = \mathbf{A}[T\_, A]$, *has a left adjoint, and is then this left adjoint.*

Having a morphism from $Lan_Y TF$ to A (where F and A are arbitrary objects of $\mathbf{Set}^{\mathbf{M}^{op}}$ and $\mathbf{A}$, respectively) amounts to having a cone λ from $T\pi$ to A. Such a cone is indexed by the pairs $(M, \mu : YM \to F)$, or equivalently (by Yoneda lemma!) by the pairs (M, x) such that $x \in FM$. And in turn, such a family of λ_x's describes a natural transformation from F to $\mathbf{A}[T\_, A]$. □

As a corollary of Proposition 3.3.4, we retrieve the result that every presheaf is a colimit of representable functors (Proposition 1.4.3), which amounts to say that Y is a dense functor. Indeed, this follows from the fact R_Y is (isomorphic to) the identity functor, and from the obvious fact that the left adjoint of the identity functor is the identity functor. Indeed, we have $R_Y(F)C = \mathbf{Set}^{\mathbf{M}^{op}}[YC, F]$, and hence $R_Y(F)C \cong FC$ by Yoneda lemma.

We end the section by examining some properties of dense functors.

Proposition 3.3.5 *When* K *is dense, it is equivalent to have the following structures:*

1. *a left Kan extension of* T *along* K *such that* $\eta = id$;

2. *A functor* H *that preserves the colimits of all diagrams of the form* $K\pi$ *(for all objects* C *of* $\mathbf{C}$*) and is such that* $H \circ K = T$.

PROOF. $(1) \Rightarrow (2)$. We have:

$$\begin{array}{rcll} Lan_K T(\text{colim } K\pi) & = & Lan_K T(c) & \text{(by definition of density)} \\ & = & \text{colim } T\pi & \text{(by construction of } Lan_K T) \\ & = & \text{colim } ((Lan_K T \circ K)\pi) & \text{(since } \eta = id \text{ implies a fortiori} \\ & & & Lan_K T \circ K = T) \end{array}$$

More precisely, we must check that the obvious cone of vertex C is mapped by $Lan_K T$ to the colimiting cone $(Lan_K Tc, \lambda)$, i.e., that $\lambda_f = Lan_K Tf$ for all $f = C \to KM$, and this follows from the assumption $\eta = id$ and from Equation 3.1.

$(2) \Rightarrow (1)$ Applying H to the obvious cone of vertex C, we obtain a colimit of $T\pi = HK\pi$, with vertex HC, by our assumptions on K and H. These colimits give us the left Kan extension$(Lan_K T, \eta)$, with $H = Lan_K T$. Moreover, η is given by the component at id, which is $Hid = id$. □

The characterisation given in Proposition 3.3.5 holds in particular for the functors K that are dense, full, and faithful. Then the assumption $\eta = id$ holds for free in the statement, and Lan_K establishes a one-to-one correspondence between functors T from $\mathbf{M}$ to $\mathbf{A}$ and functors H from $\mathbf{C}$ to $\mathbf{A}$ that preserve the colimits of the form

$K\pi$. Indeed, if $Lan_K T_1 = Lan_K T_2$, then $T_1 = (Lan_K T_1) \circ K = (Lan_K T_2) \circ K = T_2$, and for every H we have $H = Lan_K(HK)$.

When $\mathbf{C} = \mathbf{Set}^{\mathbf{M}^{op}}$ and $K = Y$, then $Lan_K T$ preserves all exisitng colimits (since left adjoints preserve colimits). Since $\mathbf{Set}^{\mathbf{M}^{op}}$ has all (small) colimits, $Lan_K T$ preserves all (small) colimits. Therefore the above one-to-one correspondence, in this case, can be rephrased as a one-to-one correspondence between functors T from $\mathbf{M}$ to $\mathbf{A}$ and functors H from $\mathbf{C}$ to $\mathbf{A}$ that preserve all colimits. In other words (and with a little extra work), the functor $\mathbf{Set}^{-^{op}}$ is left adjoint to the forgetful functor from **CoComp** to **Cat**, with unit Y, where **CoComp** is the category of cocomplete categories (i.e. that have alll colimits) and functors that preserve all colimits.

Exercise 3.3.6 *Show that every right adjoint preserves all right Kan extensions.*

Chapter 4

Algebras, coalgebras, bialgebras

4.1 Free versus initial

Let us recall from Section 1.5 that, given a category $\mathbf{C}$ and a functor $\Sigma : \mathbf{C} \to \mathbf{C}$ (viewed as an abstract signature), a Σ-algebra is a pair (A, a), where A is an object of $\mathbf{C}$ (called the carrier of the algebra) and a is a morphism from ΣA to A, and that a Σ-algebra morphism between two algebras (A, a), (B, a) is a morphism from A to B that preserves the algebra structure. In symbols, we have a category $\mathbf{C}^\Sigma$ whose objects are the Σ-algebras, and whose homsets are defined by the following formula:

$$\mathbf{C}^\Sigma[(A, a), (A', a')] = \{f : a \to a' \mid f \circ a = a' \circ \Sigma f\}$$

We recall that the free algebra functor T_Σ associated with Σ is defined as left adjoint to the forgetful functor U such that $U(A, a) = A$ and $Uf = f$. We shall give an equivalent description of T_Σ. We set $T_\Sigma(A) = (TA, \nu : \Sigma TA \to TA)$. By definition of an adjunction, for all A there exists a morphism $\eta_A : A \to TA$ (the unit) such that, for all X, for all Σ-algebra (A, a), and for all morphism $f : X \to A$, there exists a unique morphism, which we shall denote as $[f, a]^\#$, such that the following diagram commutes:

$$\begin{tikzcd}
X \arrow[r, "\eta_X"] \arrow[dr, "f"'] & TX \arrow[d, "{[f,a]^\#}"] & \Sigma TX \arrow[l, "\nu_X"'] \arrow[d, "{\Sigma([f,a]^\#)}"] \\
 & A & \Sigma A \arrow[l, "a"]
\end{tikzcd}$$

The left triangle is the familiar adjunction triangle (with the unit), while the right square expresses the fact that the extension $[f, a]^\#$ is a morphism of Σ-algebras. If $\mathbf{C}$ has coproducts, we can write the diagram equivalently as follows:

$$\begin{tikzcd}
X + \Sigma TX \arrow[r, "{[\eta_X, \nu_X]}"] \arrow[d, "{id + \Sigma([f,a]^\#)}"] & TX \arrow[d, "{[f,a]^\#}"] \\
X + \Sigma A \arrow[r, "{[f,a]}"] & A
\end{tikzcd}$$

or, setting $\Sigma_X(Y) = X + \Sigma Y$:

$$\begin{tikzcd}
\Sigma_X(X) \arrow[r, "{[\eta_X,\nu_X]}"] \arrow[d, "{\Sigma_X([f,a]^{\#})}"] & TX \arrow[d, "{[f,a]^{\#}}"] \\
\Sigma_X(A) \arrow[r, "{[f,a]}"] & A
\end{tikzcd}$$

We have thus expressed the adjunction $T_\Sigma \dashv U$ in terms of initial algebras: U has a left adjoint if and only if all functors Σ_X have an initial Σ_X-algebra.

When these datas exist, we have a monad on $\mathbf{C}$, which we denote simply by T, i.e., $T = UT_\Sigma$. We spell out the definition of the multiplication μ of this monad. Recall that when a monad is induced by an adjunction $F \dashv G$, its multiplication is defined by $\mu_A = G\epsilon_{FA}$. So we first spell out the definition of the counity of the adjunction. We have $\epsilon_{(A,a)} = [id_A, a]^{\#}$:

$$\begin{tikzcd}
A \arrow[r, "\eta_A"] \arrow[dr, "id"'] & TA \arrow[d, "\epsilon_{(A,a)}"] & \Sigma TA \arrow[l, "\nu_A"'] \arrow[d, "\Sigma\epsilon_{(A,a)}"] \\
& A & \Sigma A \arrow[l, "a"]
\end{tikzcd} \tag{4.1}$$

Thus, μ_A is the unique arrow that makes the following diagram commute:

$$\begin{tikzcd}
TA \arrow[r, "\eta_{TA}"] \arrow[dr, "id"'] & TTA \arrow[d, "\mu_A"] & \Sigma TTA \arrow[l, "\nu_{TA}"'] \arrow[d, "\Sigma\mu_A"] \\
& TA & \Sigma TA \arrow[l, "\nu_A"]
\end{tikzcd}$$

We now revisit the concrete case of a signature. Let $\Sigma Y = Y + (Y \times Y)$ (corresponding to the signature $\Sigma = \{a, f\}$ of Section 1.5). The initial Σ_X-algebra is the algebra of terms generated by the following syntax:

$$t ::= x \mid a \mid f(t,t)$$

The function η_X "is" the variable case, and ν_X corresponds to the other cases of the syntax. The function $[f,a]^{\#}$ is defined by induction on the size of t.

We also describe the multiplication of the associated monad in concrete terms. We can write the terms of TTX with the following syntax:

$$C ::= \underline{t} \mid a \mid f(C,C)$$

The function μ_X removes the stratification (which is formalised here by underlining), i.e.:

$$\mu_X(\underline{t}) = t \qquad \mu_X(c) = c \qquad \mu_X(f(C_1,C_2) = f(\mu_X(C_1), \mu_X(C_2))$$

But these are "retrospective" verifications: we 'knew already" that the initial algebra is a term algebra. The following categorical generalisation of Kleene's fixed point theorem allows us to *synthesise* it.

Proposition 4.1.1 *If* $\mathbf{C}$ *is a category that has all* ω*-colimits, i.e., the colimits of all diagrams of the form* $\cdot \to \cdot \to^* \cdot \to \cdot \to^*$, *and if* $F : \mathbf{C} \to \mathbf{C}$ *is a functor that preserves all* ω*-colimits, then* F *has an initial algebra.*

PROOF. We shall build the initial algebra (A, a) as a colimit. The carrier of the initial algebra is the vertex of the colimiting cone (A, λ) of the diagram D formed by taking $f_1 : 0 \to F0$ (given by initiality), then $f_2 = F(f_1) : F0 \to FF0$, etc... F sends the colimit cone to a colimit of the image under F of the diagram, which is the same as D but truncated of its first morphism. Since (A, λ) is a fortiori a cone over this diagram, we can take a to be the mediating arrow from FA to A. By construction, a satisfies $a \circ Ff_n = f_{n+1}$, for all n. Let $b : FB \to B$ be another algebra. We can build a cone from D to B whose first component $g_1 : 0 \to B$ is given by initiality and whose other components are defined by induction, as follows: $g_{n+1} = b \circ Fg_n$. We are left to show that a morphism $g : A \to B$ is mediating if and only if g is a morphism of coalgebras, since the uniqueness of the mediating morphism will then translate into the uniqueness of the algebra morphism. Let $g : a \to b$, i.e., such that $g \circ a = b \circ Fg$. We show that g is mediating, i.e. that $g \circ f_n = g_n$ (by induction on n):

$$g \circ f_{n+1} = g \circ a \circ Ff_n = b \circ Fg \circ Ff_n = b \circ Fg_n = g_{n+1}$$

Conversely, if g is mediating, the same equalities show

$$g \circ a \circ Ff_n = b \circ Fg \circ Ff_n$$

for all n, and this is enough, as the components of a colimit are collectively mono (cf. Exercise 1.2.13). □

Kleene's theorem is the special case of Proposition 4.1.1 where $\mathbf{C}$ is a partial order.

It is easily checked that the concrete signature functor Σ_X preserves ω-colimits. We also check easily that $\Sigma_X 0$ is the set of variables, that $\Sigma_X(\Sigma_X 0)$ is the set formed by the variables, a, and the terms of the form $f(x, y)$. By induction, we see that $\Sigma_X^n 0$ is the set of all terms of depth n, and that the morphisms f_n are inclusions. Under these conditions, the colimit is the union of these sets. We have thus synthetised $T_\Sigma(X)$.

4.2 Freely generated monads

In this section, we relate the notion of Σ-algebra (as considered in the previous section) with that of T-algebra, as considered in Section 2.5, for $T = UT_\Sigma$. We show that there exists a bijective correspondence between Σ-algebras and T-algebras which makes the categories $\mathbf{C}^\Sigma$ and $\mathbf{C}^T$ isomorphic. The wording "T-algebra" should be understood as an abuse of language for (T, η, μ)-algebra, since one requires the T-algebras $\alpha : TA \to A$ to respect η and μ, i.e. one imposes the two equations $\alpha \circ \eta_A = id_A$ and $\alpha \circ \mu_A = \alpha \circ T\alpha$. These equations are the price to pay to go from

Σ to T. The case of concrete signatures will help to make this point clear. Given the above signature, we have the following grammars for $\Sigma(X)$ and $T_\Sigma(X)$:

$$\begin{array}{ll} \Sigma(X) & t ::= a \mid f(x,x) \\ T_\Sigma(X) & t ::= x \mid a \mid f(t,t) \end{array}$$

The first equation imposes that $\alpha(x) = x$, while the second imposes that the interpretation is compositional, i.e., writing $\alpha(f)(x,y) = \alpha(f(x,y))$:

$$\forall\, t_1, t_2 \;\; \alpha(f(t_1,t_2)) = \alpha(f)(\alpha(t_1), \alpha(t_2))$$

We first check that $\eta : id \to T$ and $\nu : \Sigma T \to T$ are natural transformations. This is a consequence of the definition of T itself as a functor, starting from the definition of adjunction in terms of the unit and of the object part of T:

$$T(f : X \to X') = [\eta_{X'} \circ f, \nu_{X'}]^\#$$

which as a drawing reads as

$$\begin{tikzcd} X \arrow[r, "\eta_X"] \arrow[d, "f"] & TX \arrow[d, "Tf"] & \Sigma TX \arrow[l, "\nu_X"'] \arrow[d, "\Sigma Tf"] \\ X' \arrow[r, "\eta_{X'}"] & TX' & \Sigma TX' \arrow[l, "\nu_{X'}"'] \end{tikzcd}$$

We shalll use the following terminology. We say that $T = UT_\Sigma$ is freely generated by Σ, and by abstraction of the special case of concrete signatures, we say that Σ is the signature functor (even if it arbitrarily chosen!), and that T is the syntactic monad (for Σ). An adjunction $F \dashv G$ is called monadic when it coincides (up to isomorphism) with the adjunction induced by the construction of the category of T-coalgebras. We have thus shown that the adjunction $T_\Sigma \dashv U$ is monadic. We are now ready to prove the claimed isomorphism.

Proposition 4.2.1 *If Σ is an endofunctor, and T is the corresponding freely generated monad, then $\mathbf{C}^\Sigma$ and $\mathbf{C}^T$ are isomorphic. This isomorphism Φ is such that*

$$\begin{tikzcd} \mathbf{C}^\Sigma \arrow[rr, "\Phi"] \arrow[dr, "U"'] & & \mathbf{C}^T \arrow[dl, "U"] \\ & \mathbf{C} & \end{tikzcd}$$

commutes (in particular, Φ is the identity on morphisms).

Proof. With a T-algebra $\alpha : TA \to A$ we associate a Σ-algebra as follows:

$$\hat{\alpha} = \alpha \circ \nu_A \circ \Sigma\eta_A$$

Conversely, we note that the counit $\epsilon_{A,a}$ of the adjunction is a morphism from $T_\Sigma U(A,a) = TA$ to A which fits. We thus set $\check{a} = \epsilon_{(A,a)} = [id_A, a]^\#$, We must first verify that $\check{a}$ is a T-algebra. The equation $(\check{a} - \eta)$ is exactly the triangle of the diagram (4.1). For the second equation, we shall write $\check{a} \circ \mu_A$ and $\check{a} \circ T\check{a}$ as two morphisms of Σ_{TA}-algebras, with the same domain and codomain, which will imply their equality by initiality. Here is the diagram for $\check{a} \circ \mu_A$:

$$
\begin{tikzcd}
TA \arrow[r, "\eta_{TA}"] \arrow[d, equal] & TTA \arrow[d, "\mu_A"] & \Sigma TTA \arrow[l, "\nu_{TA}"'] \arrow[d, "\Sigma\mu_A"] \\
TA \arrow[r, "id"] \arrow[d, equal] & TA \arrow[d, "\check{a}"] & \Sigma TA \arrow[l, "\nu_A"'] \arrow[d, "\Sigma\check{a}"] \\
TA \arrow[r, "\check{a}"] & A & \Sigma A \arrow[l, "a"']
\end{tikzcd}
$$

(the left right square commutes by definition of $\check{a}$). For $\check{a} \circ T\check{a}$, we have to change the intermediate Σ_{TA}-algebra:

$$
\begin{tikzcd}
TA \arrow[r, "\eta_{TA}"] \arrow[d, equal] & TTA \arrow[d, "T\check{a}"] & \Sigma TTA \arrow[l, "\nu_{TA}"'] \arrow[d, "\Sigma T\check{a}"] \\
TA \arrow[r, "\eta_A \circ \check{a}"] \arrow[d, equal] & TA \arrow[d, "\check{a}"] & \Sigma TA \arrow[l, "\nu_A"'] \arrow[d, "\Sigma\check{a}"] \\
TA \arrow[r, "\check{a}"] & A & \Sigma A \arrow[l, "a"']
\end{tikzcd}
$$

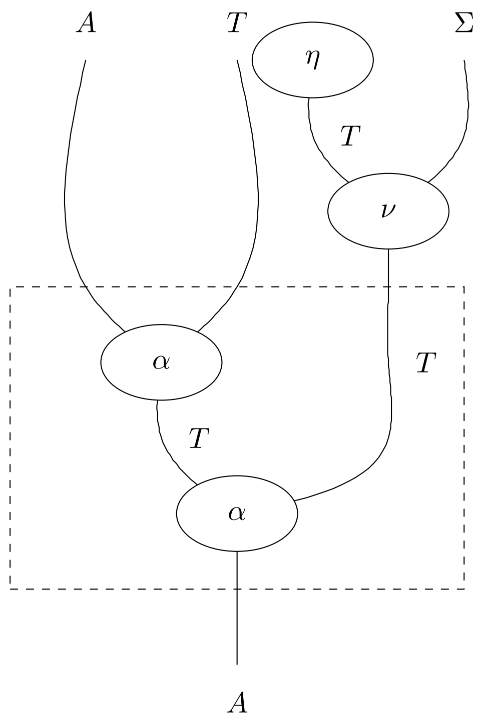
(the upper brick commutes by naturality of η and ν). We check now that the two transformations are inverse. We have:

$$\hat{\check{a}} = \underline{\check{a} \circ \nu} \circ \Sigma\eta = a \circ \Sigma\check{a} \circ \Sigma\eta = a \circ \Sigma(\check{a} \circ \eta) = a$$

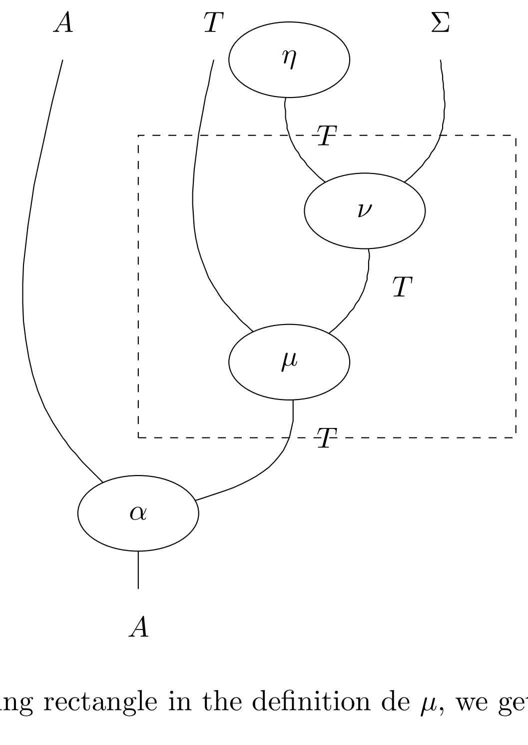
In the converse direction, by initiality it suffices to verify that α satisfies the specification of $\check{\hat{\alpha}}$, i.e.:

$$
\begin{tikzcd}
A \arrow[r, "\eta_A"] \arrow[dr, "id"'] & TA \arrow[d, "\alpha"] & \Sigma TA \arrow[l, "\nu_A"'] \arrow[d, "\Sigma\alpha"] \\
 & A & \Sigma A \arrow[l, "\hat{\alpha}"']
\end{tikzcd}
$$

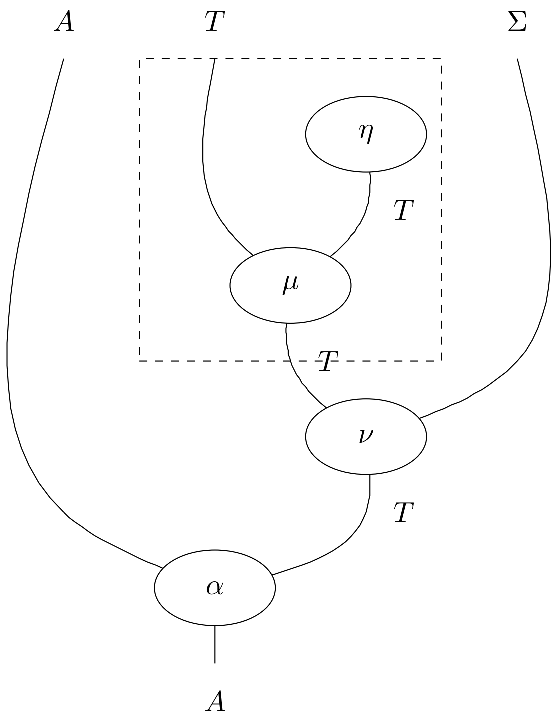
The triangle is the equation $(\alpha - \eta)$. We prove the commutation of the square by successive transformations, starting from $\hat{\alpha} \circ \Sigma\alpha$:



Applying Equation $(\alpha - \mu)$, we get:



Using the commuting rectangle in the definition de μ, we get:



and we reach $\alpha \circ \nu_A$ after an application of Equation $(\mu - \eta)$.

Finally, we need to turn $\hat{\ }$ and $\check{\ }$ into inverse functors. We show that they in fact both act as the identity on morphisms. The implication $(f : \alpha \to \beta \Rightarrow f : \hat{\alpha} \to \hat{\beta})$ follows easily from the naturality of ν and η. For the implication $(f : a \to b \Rightarrow f : \check{a} \to \check{b})$, one shows (cf. above) that $\check{b} \circ Tf$ and $f \circ \check{a}$ are two Σ_A-algebra morphisms with the same (initial) domain and codomain. □

We end the section with a variant (called the variant with accumulators) of the construction of the unique morphism from the initial algebra. The initial Σ_X-algebra is such that for all pairs (f, a) such that $f : X \to A$ and $a : \Sigma(TX \times A) \to A$ there exists a unique morphism, denoted by $[f, a]^\natural$, such that the following diagram commutes:

$$\begin{tikzcd}
X \arrow[r, "\eta_X"] \arrow[dr, "f"'] & TX \arrow[d, "{[f,a]^\natural}"] & \Sigma TX \arrow[l, "\nu_X"'] \arrow[d, "{\Sigma<id,[f,a]^\#>}"] \\
& A & \Sigma(TX \times A) \arrow[l, "a"']
\end{tikzcd}$$

It is easy to check that $\pi_2 \circ < \nu_X \circ \Sigma\pi_1, a >^\#$ fits. That the rectangle commutes is proved using the following proprerty (setting $k =< \nu_X \circ \Sigma\pi_1, h >^\#$):

$$k =< id, \pi_2 \circ k >$$

By "surjective pairing", this property is an immediate consequence of the following one:

$$\pi_1 \circ k = id$$

By initiality, this latter equality follows from the fact that π_1 is a morphism from the codomain of k to the domain of k. As for the uniqueness, we remark that if k' makes the above diagram commute, then $< id, k' >$ is a Σ_X-algebra morphism with the same domain and codomain as k, hence

$$k' = \pi_2 \circ < id, k' >= \pi_2 \circ k = \pi_2 \circ < \nu_X \circ \Sigma\pi_1, h >^{\#}$$

In the case of a concrete signature, the construction of $k' = [f, a]^{\natural}$ by induction looks like this (for f of arity 2):

$$k'(f(t_1, t_2)) = a((t_1, k'(t_1)), (t_2, k'(t_2))$$

This is the primitive recursion schema!

4.3 Coalgebras

We can make the dual constructions, starting from a functor $B : \mathbf{C} \to \mathbf{C}$, and defining $(DX, \epsilon_X, \gamma_X)$ (if it exists) as the final B_X-coalgebra, where B_X is defined by $B_X(Y) = X \times BY$. We use the following notations:

$$\begin{tikzcd} X & DX \arrow[l, "\epsilon_X"'] \arrow[r, "\gamma_X"] & BDX \\ & A \arrow[ul, "f"] \arrow[u, "<f,b>^\bullet"'] \arrow[r, "b"] & BA \arrow[u, "B<f,b>^\bullet"'] \end{tikzcd}$$

When $X = 1$, we write simply $b^\bullet$. We say that D is cofreely generated by B, that B is the behaviour functor and that D is the observational comonad (for this behaviour).

We now give concrete examples of behaviour functors. We consider first the functor

$$BX = 1 + Act \times X$$

A coalgebra $h : X \to BX$ formalises a very constrained transition system: X is a set of states. A state, if not terminal, can make a unique transition, labelled by an action a. To make this apparent, we write $(a, x') \in h(x)$ as $x \xrightarrow{a} x'$ and $* \in h(x)$ as $x \to *$ ("x is terminal"). One verifies quite easily that, for this example, DX (the comonad generated by finality) is the set of complete transition sequences, which are of two kinds:

$$\begin{array}{ll} \text{finite} & x_1 \xrightarrow{a_1} x_2 \xrightarrow{a_2} x_3 \cdots \xrightarrow{a_{n-1}} x_n \to * \\ \text{infinite} & x_1 \xrightarrow{a_1} x_2 \xrightarrow{a_2} x_3 \cdots \end{array}$$

The functions ϵ and γ are defined as follows:

$$\begin{array}{l}
\epsilon(x_1 \xrightarrow{a_1} x_2 \xrightarrow{a_2} x_3 \cdots \xrightarrow{a_{n-1}} x_n \to *) = x_1 \\
\gamma(x_1 \xrightarrow{a_1} x_2 \xrightarrow{a_2} x_3 \cdots \xrightarrow{a_{n-1}} x_n \to *) \\
\quad = (a_1, (x_2 \xrightarrow{a_2} x_3 \cdots \xrightarrow{a_{n-1}} x_n \to *)) \quad (n > 1) \\
\gamma(x \to *) = * \\
\\
\epsilon(x_1 \xrightarrow{a_1} x_2 \xrightarrow{a_2} x_3 \cdots) = x_1 \\
\gamma(x_1 \xrightarrow{a_1} x_2 \xrightarrow{a_2} x_3 \cdots) = (a_1, (x_2 \xrightarrow{a_2} x_3 \cdots))
\end{array}$$

The unique morphism $< f, b >^\bullet$ is defined as follows. We build successive prefixes π_n by induction on n:

$$\frac{}{\pi_0 = f(y)}$$

$$\frac{\pi_n = f(y) \xrightarrow{a_1} \cdots \xrightarrow{a_n} f(y') \quad b(y') = (a, y'')}{\pi_{n+1} = f(y) \xrightarrow{a_1} \cdots \xrightarrow{a_n} f(y') \xrightarrow{a} f(y'')}$$

$$\frac{\pi_n = f(y) \xrightarrow{a_1} \cdots \xrightarrow{a_n} f(y') \quad b(y') = *}{< f, b >^\bullet (y) = f(y) \xrightarrow{a_1} \cdots \xrightarrow{a_n} f(y') \to *}$$

We define $< f, b >^\bullet (y)$ as the largest (possibly infinite) π_n.

This final coalgebra construction can be synthetised using the dual of Proposition 4.1.1. If $\mathbf{C}$ has all ω^{op}-limits, i.e., the limits of the diagrams of the form

$$\cdot \leftarrow \cdot^* \leftarrow \cdot \leftarrow \cdot^* \leftarrow$$

and if $F : \mathbf{C} \to \mathbf{C}$ preserve these limits, then F has a final coalgebra, obtained by taking the limit of the diagram formed by $f_1 : F1 \to 1$ (the unique morphism to 1), $f_2 = Ff_1$ etc... Applying this construction to our example, we obtain that an element of DX is a vector $(z_1, \ldots, z_n, \ldots)$ such that $z_n = f_n(z_{n+1})$ for all n. In details, we have $z_1 = *$ (the unique element of 1), then, say, $z_2 = (x_1, (a_1, *))$. The constraint $z_2 = f_2(z_3)$ forces then z_3 to have the form $z_3 = (x_1, (a_1, (x_2, u)))$, where u is $*$ or $(a_2, *)$. If $u = *$, then the constraints $z_n = f_n(z_{n+1})$ force $z_n = z_3$ for all $n > 3$, otherwise one continues. It is clear that in this way we generate the above transition sequences.

The situation is not much different if we now take

$$BX = 1 + (Act \to X)$$

that describes a general deterministic automaton (whose terminal states do not admit any transitions). In this case, DX consists of the trees of the form

$$t = x_1 \begin{cases} \xrightarrow{a} t_1 \\ \xrightarrow{b} t_2 \\ \xrightarrow{c} t_3 \\ \vdots \end{cases}$$

with finite branching, and whose edges starting from a given node are all labelled by distinct actions. The branches of t are infinite or end with a transition $x \to *$. The morphism ϵ extracts the root of the tree, and the morphism γ extracts the immediate sub-trees of t.

The situation is more complex in presence of non-determinism. We set now

$$BX = \check{\mathcal{P}}(1 + Act \times X)$$

where $\check{\mathcal{P}}(Y)$ is the set of finite non empty subsets of Y.

A B-coalgebra is now a non deterministic transition system. The natural idea is to take the same definition as above for DX, forgetting the constraint of distinct actions. But this coalgebra, call it $D'X$ (with the corresponding morphisms ϵ' and γ'), is only *weakly* final. The above method allows us to build a morphism k_1 that fits relatively to $D'X$:

$$\begin{tikzcd}
X & D'X \arrow[l, "\epsilon_X"'] \arrow[r, "\gamma_X"] & BD'X \\
& A \arrow[ul, "f"] \arrow[u, "k_1"] \arrow[r, "b"] & BA \arrow[u, "Bk_1"']
\end{tikzcd}$$

But the morphism k_1 is not unique. The commutation of the diagram says exactly that $k_1(y)$ is a tree whose root is labelled by $f(y)$, and which is such that (neglecting * for simplicity) $b(y)$ is equal to the set of pairs (a, t) such that $k_1(y) \xrightarrow{a} t$ (i.e., t is a sub-tree whose root is related to the root of $k_1(y)$ by an edge labelled by a). But the right tree below satisfies also this specification:

$$k_1(y) = f(y) \begin{cases} \xrightarrow{a} t_1 \\ \xrightarrow{a} t_2 \\ \vdots \\ \xrightarrow{b} t_n \\ \vdots \end{cases} \qquad f(y) \begin{cases} \xrightarrow{a} t_1 \\ \xrightarrow{a} t_1 \\ \xrightarrow{a} t_2 \\ \vdots \\ \xrightarrow{b} t_n \\ \vdots \end{cases}$$

These two trees have the same immediate sub-trees, and therefore γ' does not distinguish between them. We are led to define a suitable quotient of $D'X$. Morally, one wishes to quotient by the equivalence relation generated by the contraction of two identical sub-trees with the same father. But it is more complicated than just that, due to the potential infinity of the trees. For example, we should equate

$$\cdot \xrightarrow{a} \cdot \xrightarrow{a} \cdot \ldots \cdot \xrightarrow{a} \cdot \ldots \quad \text{and} \quad \cdot \begin{cases} \xrightarrow{a} \cdot \xrightarrow{a} \cdot \ldots \cdot \xrightarrow{a} \cdot \ldots \\ \xrightarrow{a} \cdot \begin{cases} \xrightarrow{a} \cdot \xrightarrow{a} \cdot \ldots \cdot \xrightarrow{a} \cdot \ldots \\ \xrightarrow{a} \cdot \left\{ \vdots \right. \end{cases} \end{cases}$$

Note that in the second tree, there is no pair of equal subtrees with the same father. The solution is provided by the notion of bisimuation, invented by Aczel precisely for the construction of final coalgebras (in the framework of non well-founded set theory).

This complication of the construction with respect to the non-deterministic case is related to the non ω^{op}-continuity of the functor $\check{\mathcal{P}}(\_)$.

Proposition 4.3.1 *The functor $\check{\mathcal{P}}(\_)$ on* **Set** *is not ω^{op}-continuous.*

Proof. We recall that the limit of an ω^{op}-chain

$$X_0 \xleftarrow{f_1} X_1 \quad \cdots \quad X_{n-1} \xleftarrow{f_n} X_n \quad \cdots$$

is the set L of the sequences (x_n) such that $f_n(x_n) = x_{n-1}$ for all n. Take $X_0 = 1$, $f_1 = 1$ (the unique morphism to 1), $X_{n+1} = \check{\mathcal{P}}(X_n)$ and $f_{n+1} = \check{\mathcal{P}}(f_n)$. We note that there exists a sequence (x_n) in L such that the sequence $card(x_n)$ is strictly increasing. This comes from the fact that the sequence of the cardinals of the X_n's is increasing and from the fact that the functions f_n are surjective. We have thus

$$f_n(X_{n-1}) = \check{\mathcal{P}}(f_{n-1})(X_{n-1}) = X_{n-2}$$

and the sequence defined by $x_n = X_{n-1}$ fits. If $\check{\mathcal{P}}(\_)$ were continuous, $\check{\mathcal{P}}(L)$ would also be a limit of this chain. There would thus exist a cone morphism $k : L \to \check{\mathcal{P}}(L)$, and in particular the image by k of the sequence (x_n) would be a finite set $\{(y^1)_n, \ldots, (y^p)_n\}$ of sequences, of cardinal p. But the fact that k is a cone morphism imposes that, for all k, x_k is the set $\{y^1_k, \ldots, y^p_k\}$, which has a fixed cardinal. $\square$

This negative result explains why we cannot apply the general final coalgebra construction of (the dual of) Proposition 4.1.1.

We define now the notion of bisimulation, which will allow us to complete the construction of the observational comonad D. Let (A_1, f_1, b_1), (A_2, f_2, b_2) be two B_X-coalgebras. A bisimulation between b_1 and b_2 is a relation $\mathcal{R} \subseteq A_1 \times A_2$ such that if $x_1 \mathcal{R} x_2$, then:

- $f_1(x_1) = f_2(x_2)$,
- if $x_1 \to *$, then $x_2 \to *$,
- if $x_1 \xrightarrow{a} y_1$, then there exists y_2 such that $x_2 \xrightarrow{a} y_2$ and $x_2 \mathcal{R} y_2$,
- if $x_2 \to *$, then $x_1 \to *$,
- if $x_2 \xrightarrow{a} y_2$, then there exists y_1 such that $x_1 \xrightarrow{a} y_1$ and $x_1 \mathcal{R} y_1$.

On can define the notion of bisimulation for any functor B , as follows. A bisimulation between $b_1 : A_1 \to BA_1$, $b_2 : A_2 \to BA_2$ is given by a coalgebra $r : R \to BR$ and two coalgebra morphisms $\pi_1 : r \to b_1$ and $\pi_2 : r \to b_2$. Such a figure forms a span (in the category of coalgebras).

Let us check that, for our functor B (and more generally for B_X), this definition instantiates to the concrete one above. In one direction, we have to endow $\mathcal{R}$ with a coalgebra structure $r : \mathcal{R} \to \mathcal{R}$. To this aim, we set $* \in r(x_1, x_2)$ if and only if $x_1 \to *$ (and hence $x_2 \to *$), and $(x_1, x_2) \xrightarrow{a} (y_1, y_2)$ if $x_1 \xrightarrow{a} y_1$ and $x_2 \xrightarrow{a} y_2$. We take $\pi_1(x_1, x_2) = x_1$ and $\pi_2(x_1, x_2) = x_2$. We check that, say, π_1 is a coalgebra morphism. Let $x_1 \mathcal{R} x_2$. The commutation $b_1 \circ \pi_1 = B\pi_1 \circ r$ amounts to say that $x_1 \xrightarrow{a} y_1$ if and only if there exists y_2 such that $(x_1, x_2) \xrightarrow{a} (y_1, y_2)$, hence there exists y_2 such that $x_2 \xrightarrow{a} y_2$. $y_1 \mathcal{R} y_2$ and $y_1 \xrightarrow{a} y_2$.

In the other direction, if $r : R \to BR$, π_1 and π_2 are given (note that R is not necessarily a relation!), we can define a relation $\mathcal{R}$ by $x_1 \mathcal{R} x_2$ if and only if there exists $z \in R$ such that $x_1 = \pi_1(z)$ and $x_2 = \pi_2(z)$. This relation is a bisimulation: for example, if $\pi_1(z) \xrightarrow{a} u$, then, by the commutation $b_1 \circ \pi_1 = B\pi_1 \circ r$, we know that u is of the form $\pi_1(v)$, with $z \xrightarrow{a} v$, from which we deduce (using the fact that π_2 is a coalgebra morphism) that $\pi_2(z) \xrightarrow{a} \pi_2(v)$, and we have $u = \pi_1(v) \mathcal{R} \pi_2(v)$, as desired, by definition of $\mathcal{R}$.

We now consider a single coalgebra $b : A \to BA$. We write $x_1 \approx x_2$ if there exists a bisimulation $((R, r), \pi_1, \pi_2)$ between b and b, and $z \in R$ such that $x_1 = \pi_1(z)$ and $x_2 = \pi_2(z)$. In our concrete case, this amounts to say that there exists a (concrete) bisimulation $\mathcal{R}$ such that $x_1 \mathcal{R} x_2$. The relation $\approx$, which we call the bisimilarity relation, is reflexive and symmetric (immediate). It is also transitive. In the case of the above B, it follows from the easily verified property that the composition (in the sense of relations) of two bisimulations is a bisimulation (see Exercise 4.3.5 for a more general proof).

We shall need the following property. Let $b : A \to BA$ and $b' : A' \to BA'$ be two coalgebras and let $f : b \to b'$. Then we have $x \approx y$ if and only if $f(x) \approx f(y)$. The proof in the concrete case (i.e., for, say, the functor $BX = \check{\mathcal{P}}(1 + Act \times X)$) is left as an exercise. Instead, we give a general proof that illustrates the categorical definition of the notion of bisimulation. If $x \approx y$, by definition there exists $((R, r), \pi_1, \pi_2)$, $z \in R$ such that $x = \pi_1(z)$ and $y = \pi_2(z)$. Il suffices then to prolungate this span by f, on both sides: $((R, r), f \circ \pi_1, f \circ \pi_2)$ is a bisimulation (note that $f \circ \pi_1$ is a coalgebra morphism). Note that we have made use of the comfort of being able to use arbitrary spans, rather than just those arising from relations.

The other direction is more complicated. We start from $f(x) \approx f(y)$, and hence from $((R, r), \pi_1, \pi_2)$, $z \in R$ such that $f(x) = \pi_1(z)$ and $f(y) = \pi_2(z)$. We then form the pullbacks of f and π_1 (thus, the category $\mathbf{C}$ is required to have pullbacks), and of f and π_2. If $g : R_1 \to R$ (resp. $h : R_2 \to R$) is the morphism from (the vertex of) the first (resp. second) pullback to R, we form again the pullback R_3 of g and h. Then R_3 provides the carrier of the bisimulation that we need to witness that $x \approx y$. Indeed, $f(x) = \pi_1(z)$ guarantees the existence of $z_1 \in R_1$ that gets mapped to x_1 and to z (by g), and likewise there exists $z_2 \in R_2$ that is mapped to z (by h) and to x_2. Finally, since $gz_1 = hz_2$, there exists z_3 in the upper pullback that is mapped to z_1 and to z_2, and hence to x_1 and x_2.

But we are still lacking coalgebra structures on R_1, R_2, R_3, and the verification that all the morphisms that we have constructed are coalgebra morphisms. It is indeed the case, thanks to the following proposition.

Proposition 4.3.2 *If $B : \mathbf{C} \to \mathbf{C}$ sends all pulbacks to weak pullbacks, then for every span $f : b_1 \to b$ and $g : b_2 \to b$ of coalgebras, the pullback $h : A_3 \to A_1, k : A_3 \to A_2$ of f, g (in the category $\mathbf{C}$) can be completed to a commutative diagram in the category of coalgebras, i.e. there exists a coalgebra structure $b_3 : A_3 \to BA_3$ such that h and k are coalgebra morphisms. A weak pullback is defined as a pullback, omitting the uniqueness condition of the mediating morphism.*

PROOF. We show how to construct b_3. Since the figure Bf, Bg, Bh, Bk is a weak pullback, it suffices to find two morphisms h', k' from A_3 to BA_1 and BA_2, respectively, such that $Bf \circ h' = Bg \circ k'$. It is easy to see that $h' = b_1 \circ h$ and $b_2 \circ k$ fit. Finally, the commutations $Bh \circ b_3 = h'$ and $Bk \circ b_3 = k'$ say that h and k are coalgebra morphisms. □

We can now state what we have proved above.

Proposition 4.3.3 *Let $B : \mathbf{Set} \to \mathbf{Set}$. If B satisfies the assumption of Proposition 4.3.2, and if $f : (A, b) \to (A, b')$ is a coalgebra morphism, then for all x, y:*

$$x \approx y \quad \textit{if and only if} \quad f(x) \approx f(y) \ .$$

PROOF. It is easy to check that the functor $\check{\mathcal{P}}(\_)$ (and hence our example B) satisfies the required preservation condition (and does not preserve pullbacks in the strong sense). □

We can finally come back to the construction of the cofree coalgebra DX. We take $DX = D'X/\approx$. We first see that ϵ' and γ' still make sense with the quotient, and that the surjection $D'X \to DX$ defines a morphism of B_X-coalgebras. Indeed, if $t \approx t'$, then $\epsilon'(t) = \epsilon'(t')$, and we set $[t] \xrightarrow{a} [t']$ if $t \xrightarrow{a} t'$ (this does not depend on the choice of the representative of the equivalence class of t). As a consequence, we have, for any elements $[t_1]$, $[t_2]$ of DX:

$$[t_1] \approx [t_2] \quad \text{if and only if} \quad [t_1] = [t_2] \ . \tag{4.2}$$

Indeed, by Proposition 4.3.3, we have $[t_1] \approx [t_2]$ if and only if $t_1 \approx t_2$.

We set now $k(y) = [k_1(y)]$. Suppose that $k_2 : A \to DX$ is another morphism of B_X coalgebras. Then k and k_2 form a bisimulation over DX, in the abstract sense. Hence the relation $\mathcal{R}$ defined by $[t] \mathcal{R} [t_2]$ if and only if there exists $y \in A$ such that $[t] = k(y)$ and $[t_2] = k_2(y)$ is a bisimulation, in the concrete sense. Hence, by (4.2), we have $k(y) = [t] = [t_2] = k_2(y)$. We have thus the uniqueness of k, i.e., we have transformed our weakly final coalgebra $D'X$ into a final coalgebra DX.

The intimate relation between the free coalgebra construction and bisimilarity is stressed by the following result:

Proposition 4.3.4 *For any B_X-coalgebra, we have*

$$x \approx y \quad \textit{if and only if } k(x) = k(y)$$

where k is the unique coalgebra morphism to the final coalgebra DX.

Proof. By Proposition 4.3.3 and by (4.2). □

We remark finally that by the result that we have proved above, we have, for all $y_1, y_2 \in A$:

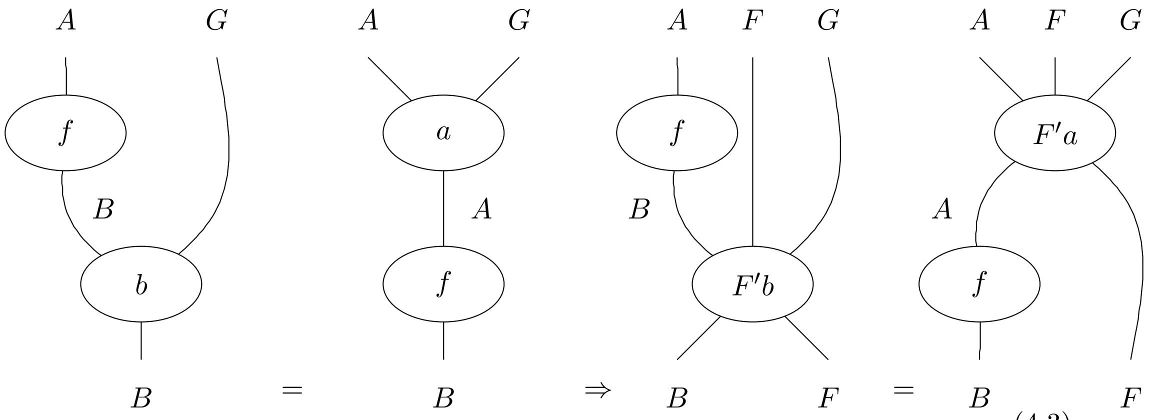
$$y_1 \approx y_2 \Leftrightarrow k_1(y_1) \approx k_1(y_2) \Leftrightarrow k(y_1) = k(y_2)$$

This justifies the terminology of observational comonad: the morphism to the final coalgebra identifies exactly the bisimilar points, i.e. the observationally equivalent points.

Exercise 4.3.5 *Show that if $B : \mathbf{Set} \to \mathbf{Set}$ satisfies the assumption of Proposition 4.3.2, then bisimilarity (with respect to B) is a transitive relation.*

4.4 Distributive laws

In this section, we are interested in the following question. Given a category $\mathbf{C}$ and two endofunctors $F : \mathbf{C} \to \mathbf{C}$ an $G : \mathbf{C} \to \mathbf{C}$, we would like to be able to define a functor $F' : \mathbf{C}^G \to \mathbf{C}^G$ such that $UF' = FU$, where $\mathbf{C}^G$ is the category of G-algebras $U : \mathbf{C}^G \to \mathbf{C}$ is the forgetful functor def (cf. Section 4.1). Such a functor is called a *functorial lifting* of F. In detail, it means that we want, for any algebra $a : GA \to A$ of carrier A, an algebra $S'a : G(FA) \to FA$ (so that $UF' = FU$ holds on objects), and such that the following implication holds, for all $f : (A, a) \to (B, b)$:

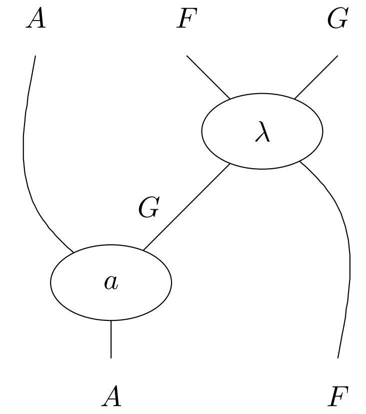


(4.3)

Note that $F'f = f$ is forced by $UF' = FU$ on morphisms. The above implication just says that F' is "correctly typed", i.e. that $F'f : F'(A, a) \to F'(B, b)$, or $F'f : F'a \to F'b$ for short.

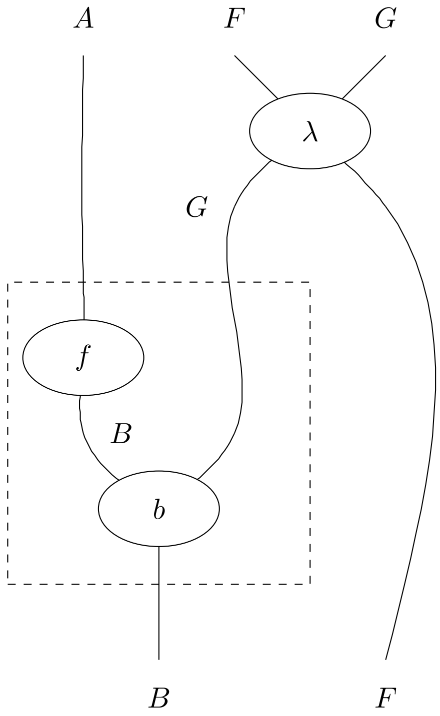
Thus, for lifting F to F', we want to find a morphism from GFA to FA, given a morphism $a : GA \to A$. Since Fa goes from FGA to FA, it seems that we would be done if we had a natural transformation $\lambda : GF \to FG$. And indeed, it works, as we show now. We proceed by steps, in a "boot-strapping" manner.

Step 1. Given $a : GA \to A$, we define $F'a = Fa \circ \lambda_A$, i.e., as



(4.4)

We show that this defines indeed a lifting $F' : \mathbf{C}^G \to \mathbf{C}^G$ of F to the category of G-algebras. We have to prove the implication (4.3). Replacing $F'b$ by its definition in $F'b \circ GFf$ we get (pushing up λ, that is, using the naturality of λ:

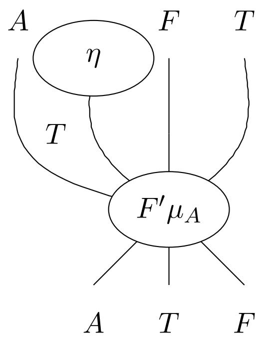


and we are done, using the assumption of the implication. What we have so far is the following.

Proposition 4.4.1 *If G, F are endofunctors on a category $\mathbf{C}$, a natural transformation $\lambda : GF \to FG$ induces a functorial lifting F' of F to the category of G-algebras, and a functorial lifting of G to the category of F-coalgebras.*

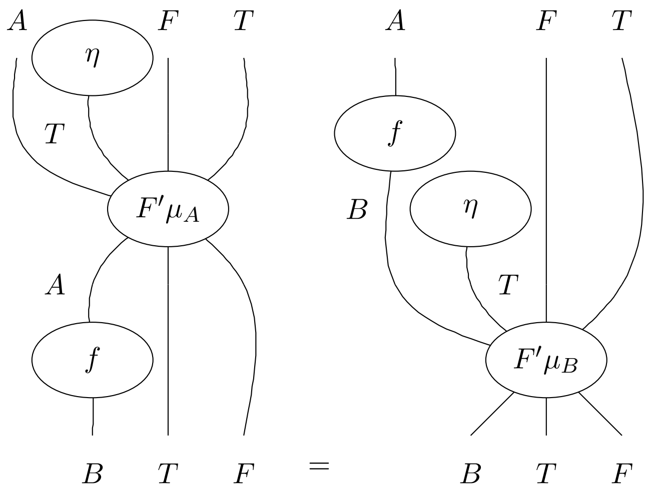
PROOF. The second part of the statement is dual (in $\mathbf{Cat}^{op}$) to the first part. $\square$

Step 2. But can we conversely define a natural transformation $\lambda : GF \to FG$, given a functorial lifting F' of F to $\mathbf{C}^G$? The answer is: yes, when G is the functor part of a monad (T, η, μ), because then we can define λ_A as follows:



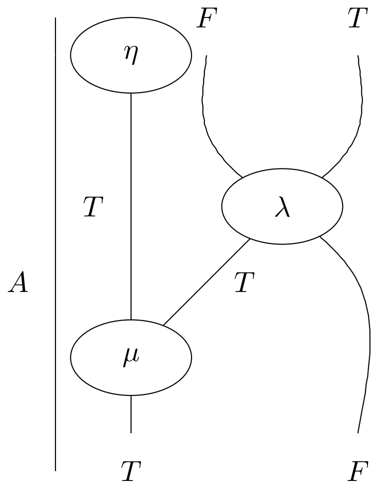
(4.5)

We check that λ is natural.

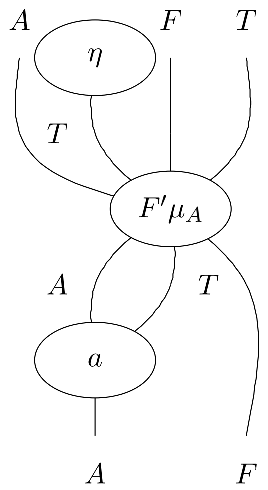


Pushing the right η up, we conclude using the implication (4.3). whose assumption is here the naturality of μ.

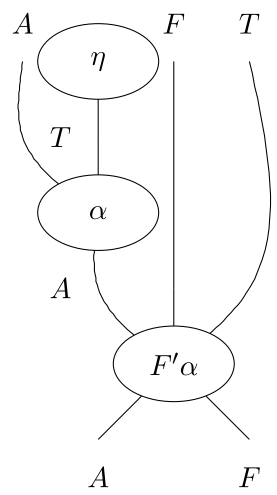
Step 3. We would like these transformations to be inverse. Starting from λ, we obtain the following natural transformation (by replacing $F'\mu_A$ by its definition in the diagram (4.5):



and we retrieve the original λ after pushing λ up and applying $(\eta - \mu)$. Conversely, starting from a lifting F', and replacing λ by its definition in the diagram (4.4), we obtain



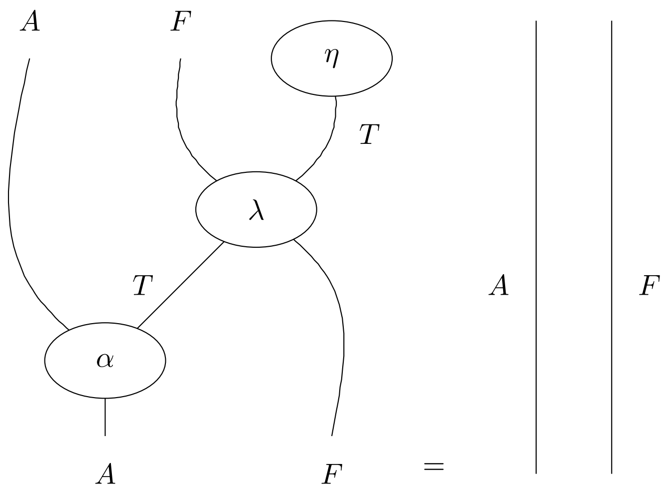
which we must equate to $F'a$. We are stuck, unless we replace $\mathbf{C}^G$ above by $\mathbf{C}^T$, the category of T-algebras, in the sense of the algebras relative to a monad, not only a functor. Then, replacing everywhere a, b by α, β, we can make use of the equation $(\alpha - \mu)$, which we can read as $\alpha : \mu_A \to \alpha$, as assumption of the implication (4.3), and we obtain (after having pushed up η):



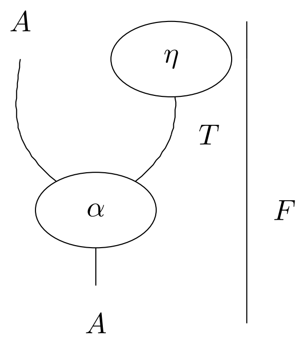
and we conclude using $(\alpha - \eta)$.

Step 4. The decision to move from the category of algebras over the underlying functor of the monad T to the category of T-algebras, whose objects respect the structure of T-algebra (which anyway was the natural thing to do) forces us to revisit step 1, since, given $\lambda : TF \to FT$, we must show that our lifting of step 1 sends a T-algebra to a T-algebra.

- Equation $(\alpha - \eta)$. We have to prove:

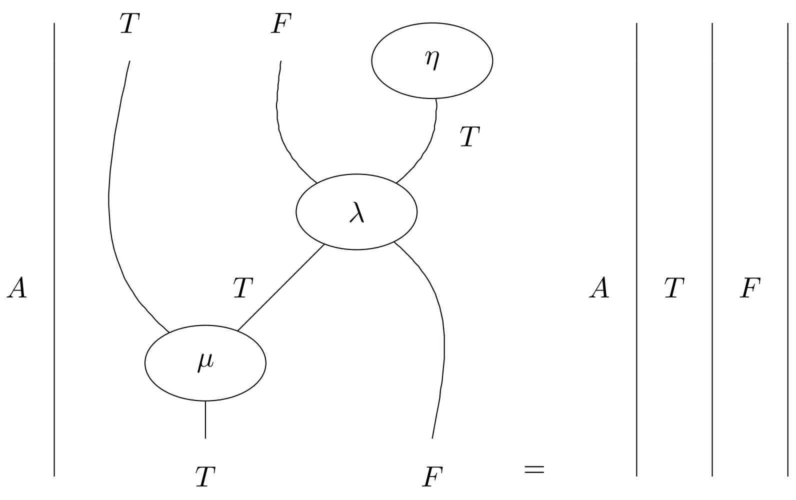


and we are stuck again, but for the last time! We shall impose two equations, $(\lambda - \eta)$ and $(\lambda - \mu)$, concerning the interaction of λ with η and μ, which the reader will find in Definition 4.4.2. Thanks to $(\lambda - \eta)$, we can transform the left hand side into



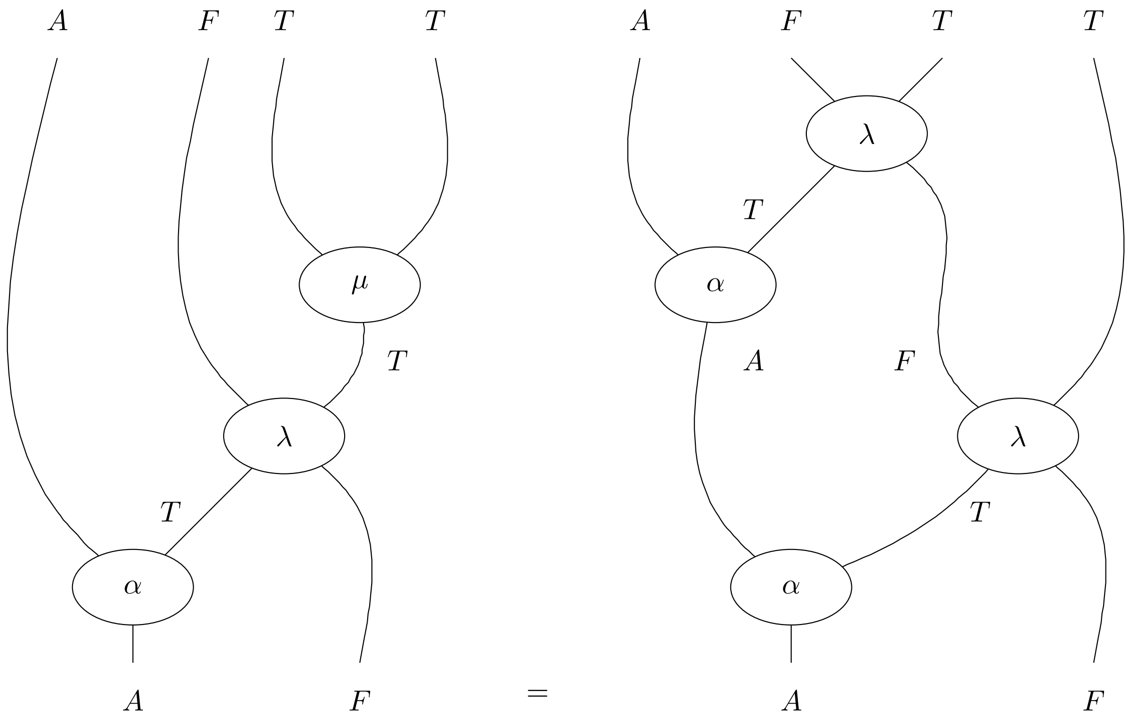
and we conclude by $(\alpha - \eta)$ applied to α.

Conversely, if $F'\alpha$ is a satisfies $(\alpha - \eta)$ for all T-algebras α, then it holds in particular for μ_A, which is a T-algebra over TA:



and, then plugging η on the top on both sides, and applying $(\eta - \mu)$ on the left (pushing λ up), we get equation $(\lambda - \eta)$.

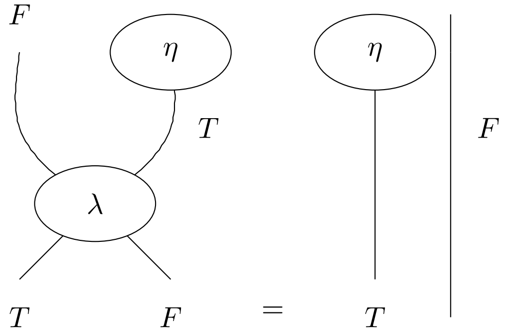
- Equation $(\alpha - \mu)$. We have to prove:



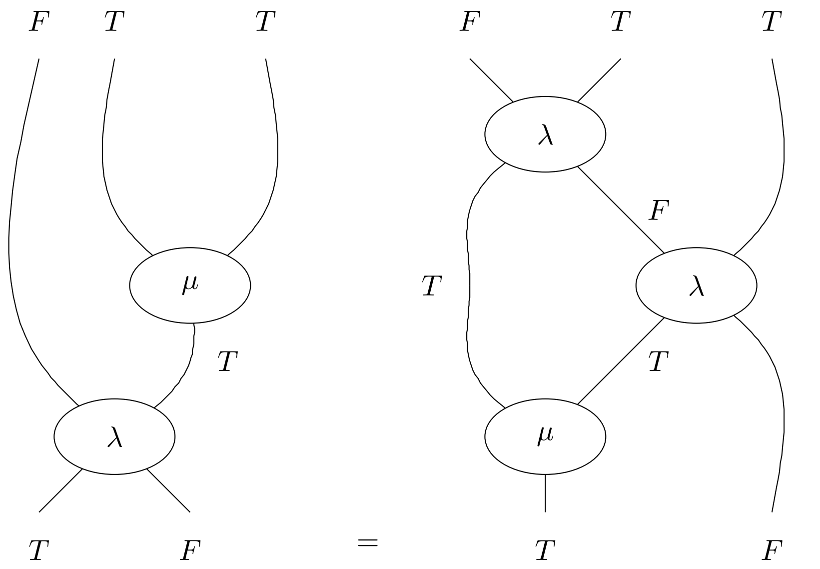
This follows by applying $(\lambda - \mu)$ followed by $(\alpha - \mu)$. Conversely, we can synthesize the equation $(\lambda - \mu)$ by specialising this equations as above with $\alpha = \mu_A$ and and by plugging an η above on both sides.

We collect the fruits of this construction in the following definition and proposition

Definition 4.4.2 *Let* $\mathbf{C}$ *be a category. A monad-functor distributive law on* $\mathbf{C}$ *is a natural transformation* $\lambda : TF \to FT$*, where* F *is an endofunctor on* $\mathbf{C}$ *and* (T, η, μ) *is a monad on* $\mathbf{C}$*, that satisfies the following two equations:*
EQUATION $(\lambda - \eta)$*:*



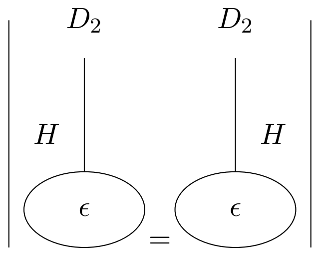
EQUATION $(\lambda - \mu)$*:*



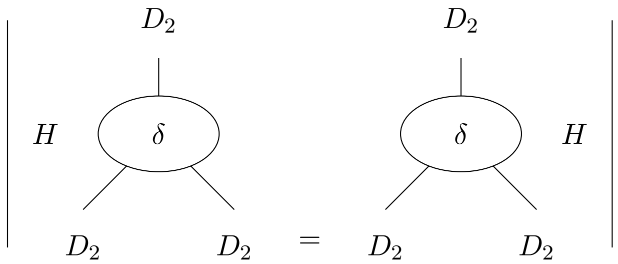
Proposition 4.4.3 *There is a one-to-one correspondence between monad-functor distributive laws $\lambda : TF \to FT$ on a category $\mathbf{C}$ and functorial liftings to the category of algebras over a monad (where T is the monad and F is the lifted endofunctor).*

We can further enrich and narrow the structure, when F is a monad S or when F is a comonad D. For example, if $F = D$ is a comonad, it is natural to require the forgetful functor $U : \mathbf{C}^T \to \mathbf{C}$ to be a comonad morphism, in the following sense (cf. the second notion of morphism of adjunctions introduced in Section 2.2.1).

Definition 4.4.4 *Let $H : \mathbf{C}_1 \to \mathbf{C}_2$ be a functor, and let D_1 and D_2 be two comonads on $\mathbf{C}_1$ and $\mathbf{C}_2$, respectively, such that $D_2 H = H D_1$. We say that H is monad morphism if it commutes with the counity and the multiplication, as follows:*



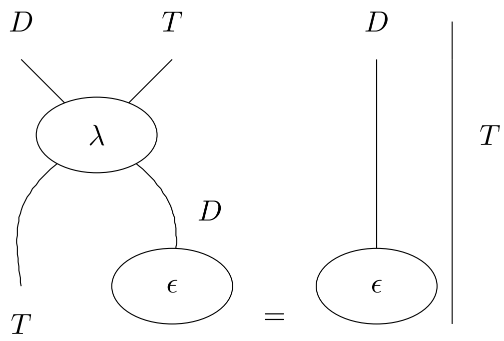
and



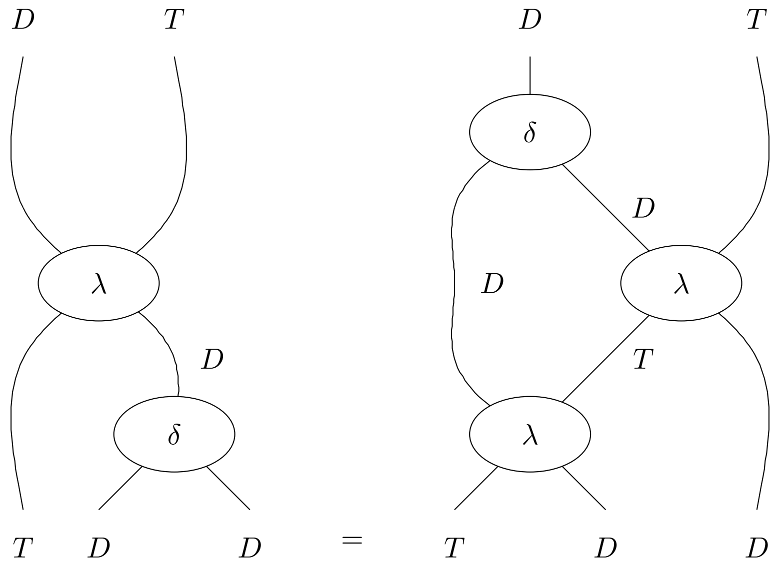
Definition 4.4.5 *Let $\mathbf{C}$ be a category.*

1. *A monad-comonad distributive law on $\mathbf{C}$ is a natural transformation $\lambda : TD \to DT$, where (D, ϵ, δ) is a comonad on $\mathbf{C}$ and (T, η, μ) is a monad on $\mathbf{C}$, which is monad-functor distributive law (for the functor underlying D) that satisfies moreover the following two equations:*

   EQUATION $(\lambda - \epsilon)$:



EQUATION $(\lambda - \delta)$:

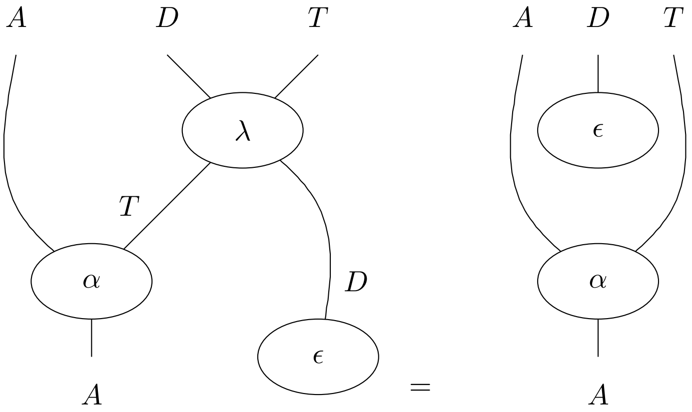


2. *A monad-monad distributive law on* $\mathbf{C}$ *is a natural transformation* $\lambda : TS \to ST$, *where* (S, η, μ) *is a monad on* $\mathbf{C}$ *and* (T, η, μ) *is a monad on* $\mathbf{C}$, *that satisfies moeover two equations* $(\eta - \lambda)$ *and* $(\mu - \lambda)$, *similar to the equations* $(\lambda - \eta)$ *and* $(\lambda - \mu)$, *except now that in the left hand side, the* η *and the* μ *are now plugged on the left, on top of* S.

Proposition 4.4.6 *1. There is a bijective correspondence between monad-comonad distributive laws* $\lambda : TD \to DT$ *on a category* $\mathbf{C}$ *and comonad liftings to the category of algebras over a monad (where* D *is the lifted comonad), where by comonad lifting we mean a functorial lifting for which* $U : \mathbf{C}^T \to \mathbf{C}$ *becomes a comonad morphism.*

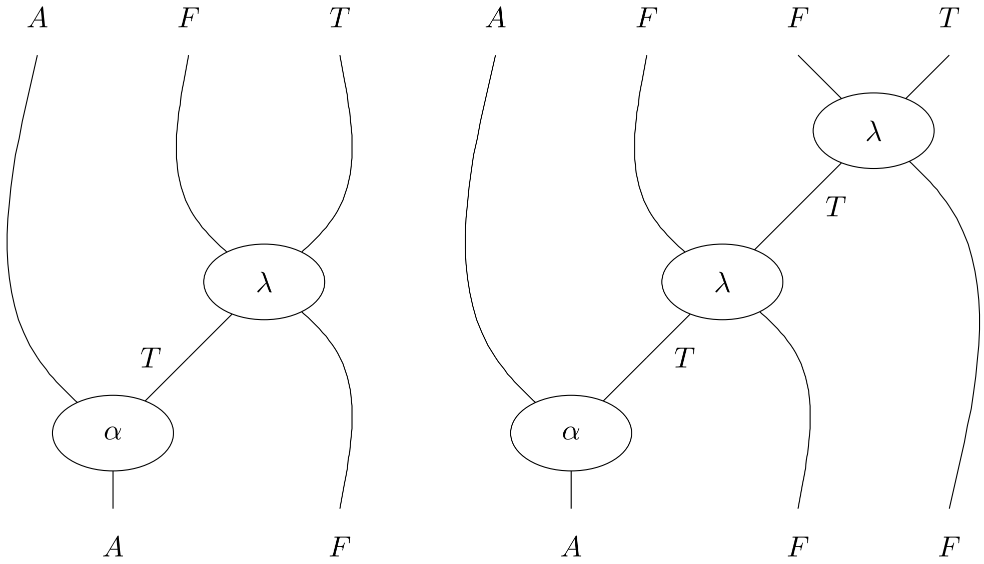
2. *Likewise there is a bijective correspondence between monad-monad distributive laws* $\lambda : TS \to ST$ *on a category* $\mathbf{C}$ *and monad liftings to the category of algebras over a monad (where* S *is the lifted monad).*

Proof. We prove only (1), the proof of (2) being similar. Suppose that $(\lambda - \epsilon)$ holds. We have to prove that $\epsilon_A : D'\alpha \to \alpha$ for any T-algebra α, i.e.:



But $(\lambda - \epsilon)$ does precisely this for us. Moreover, like in the proof of Proposition 4.4.3 (step 4), we can conversely synthesise $(\lambda - \epsilon)$ by specialising this equality to $\alpha = \mu_A$.

The commutation of δ with U is handled likewise. The following simple type-matching should convince the reader that the proposition indeed works both for $F = D$ and $F = S$. We have on one side $F'(\alpha)$ and on the other side $F'(F'(\alpha))$, as follows:



Then observe that in order to match the interfaces (one top F versus two top F, one bottom F versus two bottom F), we have the choice of:

1. having on both sides one top F and two bottom F', with the help of a comultiplication δ: this is the monad-comonad case, with $F = D$.

2. having on both sides two top F's and one bottom F, with the help of a multiplication μ: this is the monad-monad case, with $F = S$. $\square$

By dualisation, we obtain the following two new notions (only two, because the notion of monad-comonad distributive law is self-dual): corresponding to the self-dual notion

- functor-comonad distributive laws, which are natural transformations $\lambda : GD \rightarrow$ satisfying $(\lambda - \epsilon)$ and $(\lambda - \delta)$, that are in bijective correspondence with the functorial liftings of G to the category $\mathbf{C}^D$ of D-coalgebras
- comonad-comonad distributive laws $\lambda : ED \rightarrow DE$ which satisfy moreover two equations $(\epsilon - \lambda)$ and $(\delta - \lambda)$, that are in bijective correspondence with the comonad liftings of E.

For the monad-comonad case, we get another bijective correspondence:

- Monad-comonad distributive laws $\lambda : TD \rightarrow DT$ are in bijective correspondence with the monad liftings of T to the category $\mathbf{C}^D$ of D-coalgebras.

The only combination that does not work is comonad-monad, because we need G to be a monad or F to be a comonad to get a bijective correspondence with a notion of lifting.

We make two final observations. In the proof of Proposition 4.4.6, when proving that $(D \mapsto D')$ is a comonad lifting, we did not make use of the monad structure of T. We thus have the following enrichment of Proposition 4.4.1.

Proposition 4.4.7 *1. A functor-comonad distributive law $\lambda : GD \rightarrow DG$ induces induces a comonad lifting D' of D to the category of G-algebras.*

2. A monad-functor distributive law $\lambda : TF \rightarrow FT$ induces a monad lifting T' of T to the category of F-coalgebras.

Proof. The second part of the statement follows from the first by duality. $\square$

Our second observation is that if G, B are two endofunctors on $\mathbf{C}$, in such a way that there is a functorial lifting of G to $G' : \mathbf{C}^B \rightarrow \mathbf{C}^B$, where $\mathbf{C}^B$ is the category of B- coalgebras (the letter B, for "behaviour", witnesses that we are taking coalgebras and not algebras), then, by Proposition 4.2.1, we have also a functorial lifting of G to the the category of D-coalgebras, where D is the freely generated comonad associated with B. If moreover $G = T$ is a monad, and if the initial lifting is in fact a monad lifting (i.e. $G' = T'$ is a monad, and the forgetful functor commutes with the unities and multiplications), then the lifting of T to the category of D-coalgebras is also a monad lifting. This is clear, since the monad lifting property says that the unit and multiplication are B-algebra morphisms, and hence D-algebra morphisms since the isomorphism between $\mathbf{C}^B$ and $\mathbf{C}^D$ is the identity on objects. We summarize (the second part of) this second observation in the following statement.

Proposition 4.4.8 *Let* $\mathbf{C}$ *be a category. If* T *is a monad on* $\mathbf{C}$*, and if* B *is an endofunctor of* $\mathbf{C}$ *with associated cofreely generated comonad* D*, then the following are equivalent:*

1. *there exists a monad lifting of* T *to* $\mathbf{C}^B$*,*
2. *there exists a monad lifting of* T *to* $\mathbf{C}^D$*.*

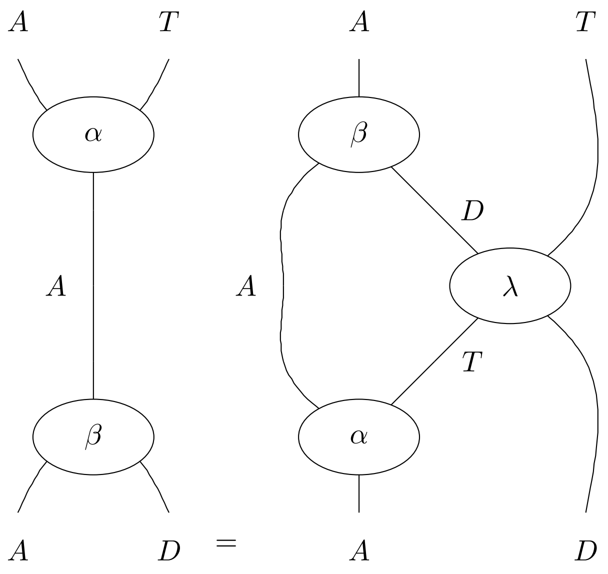
Bialgebras. We now concentrate on the monad-comonad case. We suppose thus that we have one of the following equivalent structures relating a monad T and a comonad D on a category $\mathbf{C}$:

1. a monad-comonad distributive law $\lambda : TD \to DT$,
2. a monad lifting T' of T to $\mathbf{C}^D$ (D-coalgebras),
3. a comonad lifting D' to $\mathbf{C}^T$ (the T-algebras).

It is quite natural to consider then objects A carrying both a structure of T-algebra and a structure of D-coalgebra. This leads us to the following definition.

Definition 4.4.9 *A bialgebra is a triple* (A, α, β) *where* (A, α) *is a* T*-algebra and* (A, β) *is a* D*-coalgebra that satisfies one of the following three equivalent conditions:*

1. *the following equation is satisfied:*



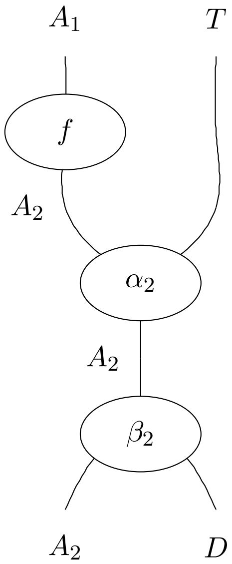
(4.6)

2. $\alpha : T'\beta \to \beta$ *is an algebra morphism,*
3. $\beta : \alpha \to D'\alpha$ *is a coalgebra morphism.*

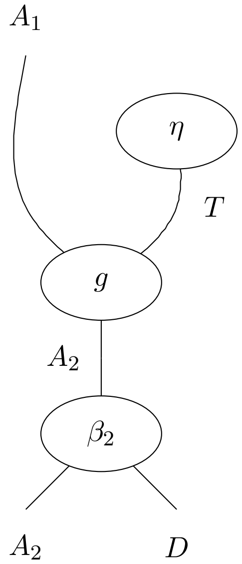
A morphism of bialgebras is a morphim that is both an algebra and coalgebra morphism. We write $\mathbf{C}^\lambda$ for the category of bialgebras and bialgebra morphisms.

Proposition 4.4.10 *The forgetful functor $\Pi' : \mathbf{C}^\lambda \to \mathbf{C}^D$ that maps (A, α, β) to (A, β) and acts as identity on morphisms has a left adjoint F^λ, defined on objects by $F^\lambda(A, \beta) = (TA, \mu_A, \lambda_A \circ T\beta)$.*

Proof. Let us fix (A_1, β_1) and (A_2, α_2, β_2). Let $f : (A_1, \beta_1) \to (A_2, \beta_2)$ and $g : (TA_1, \mu_{A_1}, \lambda_{A_1} \circ T\beta_1)$ Adapting what we have done in Section 2.5.1, we define $\zeta(g) = g \circ \eta_{A_1}$ and $\xi(f) = \alpha_2 \circ Tf$. We already know from there that the transformations are inverse. We just need to wheck that $\zeta(g)$ and $\xi(f)$ are coalgebra morphisms. For $\xi(f)$, starting from the string diagram for $\beta_2 \circ \xi(f)$,



we apply Equation (4.6), and then the equation expressing the assumption that $f : (A_1, \beta_1) \to (A_2, \beta_2)$ is a coalgebra morphism, and we obtain $D(\xi(f)) \circ (\lambda_{A_1} \circ T\beta_1)$ as desired. For $\zeta(g)$, starting from the string diagram for $\beta_2 \circ \zeta(g)$,



we apply the equation expressing the assumption that $g : (TA_1, \lambda - A_1 \circ T\beta_1)$ is a coalgebra morphism, and then Equation $(\lambda - \eta)$, and we obtain $D(\zeta(g)) \circ \beta_1$ as desired. $\square$

Proposition 4.4.11 *The forgetful functor $\Pi : \mathbf{C}^\lambda \to \mathbf{C}^T$ that maps (A, α, β) to (A, α) and acts as identity on morphisms has a right adjoint G_λ, defined on objects by $G_\lambda(A, \alpha) = (DA, D\alpha \circ \lambda_A, \delta_A)$.*

Proof. This statement is dual to the statement of Proposition 4.4.10. $\square$

Corollary 4.4.12 *Assuming that $\mathbf{C}$ has a terminal object, the category $\mathbf{C}^\lambda$ has a terminal object, namely $G^\lambda(1, alpha)$, where α is the unique morphism from $T1$ to 1.*

Proof. It is immediate to check that $(1, \alpha)$ is a terminal T-algebra. The conclusion follows from the fact that right adjoints preserve limits. $\square$

Likewise, if $\mathbf{C}$ has an initial object, then $\mathbf{C}^\lambda$ has an initial object.

4.5 Bialgebras and operational semantics

We consider the behaviour functor:

$$BX = (\mathcal{P}_f X)^{Act}$$

where $\mathcal{P}_f$ denotes the finite powerset functor. It is a simplified variant of the behaviour $BX = \check{\mathcal{P}}(1 + Act \times X)$ of Section 4.3 (note that $(\mathcal{P}_f X)^{Act} \approx \mathcal{P}_f(Act \times X)$).

We consider the following mini-langage (a fragment of CCS):

$$t ::= x \mid \mathtt{nil} \mid a.t \mid (t|t)$$

whose operational semantics is specified by the following rules:

$$\frac{}{a.x \xrightarrow{a} x} \qquad \frac{x_1 \xrightarrow{a} x_1'}{x_1|x_2 \xrightarrow{a} x_1'|x_2} \qquad \frac{x_2 \xrightarrow{a} x_2'}{x_1|x_2 \xrightarrow{a} x_1|x_2'}$$

These rules allow us to construct proof trees of judgements of the form $P \xrightarrow{a} Q$, i.e., they define a B-coalgebra of carrier $T0$ (where 0 is the initial object of **Set**, i.e., the empty set). More generally, if we are given a B–coalgebra $\beta : A \to BA$, and if we add the following rules to the ones above:

$$\frac{v \in \beta(a)(u)}{u \xrightarrow{a} v}$$

we can define a coalgebra $\overline{\beta} : TA \to BTA$ by setting $Q \in \overline{\beta}(a)(P)$ if and only if $P \xrightarrow{a} Q$ is provable by the rules of our fragment of CCS plus the rules imported from β. In particular, choosing $A = 0$, β is necessarily the morphism given by initiality and there is nothing to import.

Theorem 4.5.1 *The bisimilarity on the coalgebra $\overline{\beta} : TA \to BTA$ is a congruence.*

In particular, taking $A = 0$, we have that bisimilarity for this fragment of CCS is a congruence, i.e., if$t \approx t'$, then $a.t \approx a.t'$ and if $t_1 \approx t_1'$ et $t_2 \approx t_2'$, then $t_1|t_2 \approx t_1'|t_2'$.

Its proof of this important result will take profit of the material accumulated so far: the characterization of bisimulation given in Section 4.3, the enrichment of a coalgebra structure into a bialgebra structure provided by Proposition 4.4.10, and the final bialgebra construction provided by Corollary 4.4.12. We thus embark into showing how the transformation that maps β to $\overline{\beta}$ can be synthetised.

The CCS rules induce a transformation

$$\rho : \Sigma(id \times B) \to BT$$

where Σ is the signature functor associated with the syntax, i.e.

$$\Sigma X = 1 + X + \dots X + X \times X$$

where there are as many copies of X as there are actions in *Act*. The transformation $\rho_A = [\rho_{\mathtt{nil}}, \dots, \rho_{a.}, \dots, \rho_|] : \Sigma(A \times BA) \to BTA$ is defined as follows:

$$\begin{array}{l} \rho_{\mathtt{nil}}(a) = \emptyset \ \ \text{(for all } a) \quad \rho_{a.}(y, \beta)(b) = \left\{ \begin{array}{ll} \{y\} & (b = a) \\ \emptyset & (b \neq a) \end{array} \right. \\ \rho_|((y_1, \beta_1), (y_2\beta_2))(a) = \{y_1'|y_2 \mid y_1' \in \beta_1(a)\} \cup \{y_1|y_2' \mid y_2' \in \beta_2(a)\} \end{array}$$

This way of defining ρ is systematic. We suppose that for each operator f of arity n of a signature Σ we are given a finite set of rules of the form

$$\frac{x_1 \xrightarrow{a_1} X_1^{a_1} \ \ x_1 \xrightarrow{a_2} X_1^{a_2} \ \ \dots \ \ x_n \xrightarrow{a_1} X_n^{a_1} \ \ \dots}{f(x_1, \dots, x_n) \xrightarrow{c} t}$$

where, for exemple, $x_1 \xrightarrow{a_1} X_1^{a_1}$ is an abbreviation for $\{x_1 \xrightarrow{a_1} x_1' \mid x_1' \in X_1^{a_1}\}$, and where:

1. the X_i^a's are pairwise disjoint finite sets of variables, disjoint from $\{x_1, \dots x_n\}$;
2. the variables appearing in t appear in the union of the X_i^a's and of $\{x_1, \dots x_n\}$.

For all A, we define $\rho_f : (A \times BA)^n \to BTA$ (and hence, globally, $\rho_A = [\rho_f, \dots])$ as follows: $\rho_f((y_1, \beta_1), \dots, (y_n, \beta_n))(c)$ is the set of the terms $\rho(t)$ such that

- ρ is a substitution defined on the set of variables appearing in the rules that define the operational semantics of f, and such that $\rho(x_1) = y_1, \dots, \rho(x_n) = y_n$;
- there exists a rule for f such that $\rho(X_1^{a_1}) \subseteq \beta_1(a_1)$, $\rho(X_1^{a_2}) \subseteq \beta_1(a_2)$,$\dots$, $\rho(X_n^{a_1}) \subseteq \beta_n(a_1)$,$\dots$.

The fundamental property of ρ_f (and hence of $\rho_{\texttt{nil}},\ldots,\rho_{a.},\ldots,\rho_{|}$, and hence of ρ) is the following:

$$\rho_f \text{ is a natural transformation.}$$

Let $h : A \to C$. We have to verifiy the following equality: $BTh \circ \rho_f = \rho_f \Sigma(f \times Bf)$:

- $(BTh \circ \rho_f)((y_1,\beta_1),\ldots,(y_n,\beta_n))(a)$ is the set of the $(\sigma \circ \rho)(t)$'s, where σ is the substitution that maps each element $y \in A$ to $h(y) \in C$;
- $\Sigma(f \times Bf)((y_1,\beta_1),\ldots,(y_n,\beta_n)) = (h(y_1),\beta_1'),\ldots,h(y_n),\beta_n')$, where $\beta_i'(a) = h\beta_i(a))$.

We have to compare the set of the $(\sigma \circ \rho)(t)$'s with the set of the $\rho'(t)$,'s where $\rho'(x_1) = h(y_1)$,..., $\rho'(x_n) = h(y_n)$, $\rho'(X_1^{a_1}) \subseteq h(\beta_1(a_1))$, $\rho(X_1^{a_2}) \subseteq h(\beta_1(a_2))$,..., $\rho(X_n^{a_1}) \subseteq h(\beta_n(a_1))$,.... Clearly, every $\sigma\rho$ is a ρ' that fits. Conversely, we have to show that every ρ' that fits can be factorised as $\rho = \sigma \circ \rho'$. We construct ρ as follows:

- for each x_i, we set $\rho(x_i) = y_i$ (impose);
- for each X_i^a, since $\rho'(X_i^a) \subseteq h(\beta_i(a))$, for each $x' \in X_i^a$) we can choose an element y of $\beta_i(a)$ such that $h(y) = \rho'(x)$, and set $\rho(x') = y$. These choices do not interfer with those made on $x_1,\ldots,x_n$ and on the other X_i^a's, by condition (1).
- The function ρ' does not need to be defined elsewhere, by condition (2) (note that if we had allowed t to contain another variable z, we could then choose $\rho'(z)$ outside the image of h, and hence ρ' would not factorise through A).

We show now how ρ allows us to define the transformation that associates $\overline{\beta}$ to β.

- We first define
$$\rho' = B\mu \circ \rho : \Sigma(T \times BT) \to BT$$
This transformation formalises the *instantiation* of a rule. For example, the instantiation of the first $|$ rule with t_1, t_1', t_2' yields the term $t_1'|t_2 = \mu(\underline{t_1'}|\underline{t_2})$, where the underlining recalls that ρ_{TA} goes to $BTTA$.
- Then we obtain $\overline{\beta}$ by (the variant with accumulators of) initiality:

$$\begin{tikzcd}
A \arrow[r, "\eta_A"] \arrow[dr, "B\eta_A \circ \beta"'] & TA \arrow[d, "\overline{\beta}"] & \Sigma TA \arrow[l, "\nu_A"'] \arrow[d, "\Sigma<id,\overline{\beta}>"] \\
 & BTA & \Sigma(TA \times BTA) \arrow[l, "\rho'_A"']
\end{tikzcd}$$

This diagram formalises the inductive construction of a proof tree by *repeated* application of the (instantiated) rules.

From now on, we can ignore syntactic details, Our only assumptions are that we have the following structures available:

- a category $\mathbf{C}$ with products,
- an endofunctor Σ on $\mathbf{C}$ that freely generates a monad T,
- an endofunctor B on $\mathbf{C}$,
- and a natural transformation $\rho : \Sigma(id \times B) \to BT$.

As we have just seen, these data allow us to define an operation $\beta \mapsto \overline{\beta}$ mapping a coalgebra (A, β) to a coalgebra having TA as carrier. We shall show that this constitutes a monad lifting of T to $\mathbf{C}^B$ (cf. Section 4.4). For convenience, we recall the properties that we have to prove:

1. for every $f : \beta_1 \to \beta_2$, we have $Tf : \overline{\beta_1} \to \overline{\beta_2}$;
2. for every coalgebra $\beta : A \to BA$, η_A and μ_A are coalgebra morphisms, i.e.

$$\eta_A : \beta \to \overline{\beta} \qquad \mu_A : \overline{\overline{\beta}} \to \overline{\beta}$$

We establish (1) by showing that under the assumption $Bf \circ \beta_1 = \beta_2 \circ f$, $BTf \circ \overline{\beta_1}$ and $\overline{\beta_2} \circ Tf$ satisfy the same universal property:

$$\begin{tikzcd}
A_1 \arrow[r, "\eta_{A_1}"] \arrow[dr, "B\eta_{A_2} \circ \beta_2 \circ f"'] & TA_1 \arrow[d, "-"] & \Sigma TA_1 \arrow[l, "\nu_{A_1}"'] \arrow[d, "\Sigma<id,_>"] \\
& BTA_2 & \Sigma(TA_1 \times BTA_2) \arrow[l, "\rho'_{A_2} \circ \Sigma(Tf \times id)"']
\end{tikzcd}$$

- For $BTf \circ \overline{\beta_1}$, the commutation of the triangle follows from the commutation of the triangle relative to $\overline{\beta_1}$, from the hypothesis on f and from the naturality of η, and the commutation of the rectangle follows from the commutation of the rectangle relative to $\overline{\beta_1}$ and (modulo the reorganisation $\Sigma(Tf \times BTf) = \Sigma(Tf \times id) \circ \Sigma(id \times BTf)$) from the naturality of ρ' (that follows itself from the naturality of ρ).
- For $\overline{\beta_2} \circ Tf$, the commutation of the triangle follows from the naturality of η and from the commutation of the triangle relative to $\overline{\beta_2}$, and the commutation of the rectangle follows from the naturality of ν and from the commutation of the rectangle relative to $\overline{\beta_2}$, modulo the reorganisation

$$\Sigma(< id, \overline{\beta_2} > \circ \Sigma Tf = \Sigma(Tf \times id) \circ \Sigma(id \times \overline{\beta_2}) \circ \Sigma(< id, Tf >)$$

We show now the property (2). For η, this is precisely the property of commutation of the triangle relative to $\overline{\beta}$. For μ, we proceed as for (1), taking now the following universal problem:

$$\begin{tikzcd}
TA \arrow[r, "\eta_{TA}"] \arrow[dr, "\overline{\beta}"'] & TTA \arrow[d, "-"] & \Sigma TTA \arrow[l, "\nu_{TA}"'] \arrow[d, "\Sigma<id,_>"] \\
& BTA & \Sigma(TTA1 \times BTA) \arrow[l, "\rho'_A \circ \Sigma(\mu \times id)"']
\end{tikzcd}$$

- $\overline{\beta} \circ \mu$: by definition of μ and modulo a rearrangement of $\Sigma(< id, \overline{\beta} >) \circ \Sigma\mu$.
- $B\mu \circ \overline{\overline{\beta}}$. By definition of $\overline{\overline{\beta}}$, we have to prove:

$$B\mu_A \circ \rho'_{TA} = \rho'_A \circ \Sigma(\mu_A \times id) \circ \Sigma(id \times B\mu_A)$$

which follows from the definition of ρ', from the naturality of ρ, and from the associativity of μ:

$$\begin{aligned} B\mu_A \circ \rho'_{TA} &= (B\mu_A \circ B\mu_{TA}) \circ \rho_{TA} \\ &= B\mu_A \circ (BT\mu_A \circ \rho_{TA}) \\ &= B\mu_A \circ \rho_A \circ \Sigma(\mu_A \times B\mu_A) \\ &= \rho'_A \circ \Sigma(\mu_A \times id) \circ \Sigma(id \times B\mu_A) \end{aligned}$$

This completes the proof of the fact that there is a monad lifting of T to the category of B-coalgebras. Hence, by Proposition 4.4.8, there exists also a monad lifting of T to the category of D-coalgebras, and this in turn is equivalent to having a distributive law $\lambda TD \to DT$. Summarizing, we have proved the following statement:

Proposition 4.5.2 *Let* $\mathbf{C}$ *be a category with products, let* $\Sigma : \mathbf{C} \to \mathbf{C}$ *with associated freely generated monad* T*, let* $B : \mathbf{C} \to \mathbf{C}$ *with associated cofreely generated comonad* D*, and let* ρ *be a natural transformation from* $\Sigma(id \times B)$ *to* BT*. Then these data induce a monad-comonad distributive law* $\lambda : TD \to DT$*.*

The precise definition of λ can be inferred from the successive transformations that we have described, but we shall not need to spell it out.

We are now ready to prove our main theorem.

PROOF OF THEOREM 4.5.1. We have to show that $\approx$ is a congruence. By Proposition 4.5.2, we have a monad-comonad distributive law. By Corollary 4.4.12, we have a final bialgebra of carrier $D1$ (with the coalgebra structure given by δ_1). By Proposition 4.4.10, we can enrich $\overline{\beta}$ (more precisely the associated D-coalgebra) into a bialgebra structure on TA (with the algebra structure given by μ_A). Therefore, we have a diagram

$$\begin{tikzcd} TD1 \arrow[r] & D1 \arrow[r] & DD1 \\ TTA \arrow[u, "Tk"] \arrow[r, "\mu_A"] & TA \arrow[u, "k"] \arrow[r] & DTA \arrow[u, "Dk"] \end{tikzcd}$$

where the right square, expressed as B-coalgebra morphism, says that k is the unique coalgebra morphism from $\overline{\beta}$ to ν_1. Therefore, by Proposition 4.3.4, if $s_1 \approx t_1$, then $k(s_1) = k(t_1)$. Let f be an operator of the signature Σ (of arity 2), and let $s_1 \approx t_1$ and $s_2 \approx t_2$. Consider $\underline{f}(s_1, s_2) \in TTA$ (the underlining of f serves to stress that we have here a term of TTA over s_1, s_2 considered as variables in TA). Since $k(s_1) = k(t_1)$ and $k(s_2) = k(t_2)$, we have (by definition of Tk)

$$(Tk)(\underline{f}(s_1, s_2)) = \underline{f}(k(s_1), k(s_2)) = (Tk)(\underline{f}(t_1, t_2))$$

Since on the other hand $\mu_A(\underline{f}(s,t)) = f(s,t)$ for all $s, t \in TA$, by commutation of the left square, we get $k(f(s_1, s_2)) = k(f(t_1, t_2))$, i.e., $f(s_1, s_2) \approx f(t_1, t_2)$, which concludes the proof. □

Chapter 5

Lambda-calculus and categories

In this chapter, we provide an introduction to the syntax of the λ-calculus (Section 5.1), and we relate it to cartesian closed categories, viewed as a syntax (Section 5.3). An intermediate syntax, which has an interest on its own, is presented in Section 5.2. Finally, the connection between λ-calculus and cartesian closed categories is exploited in Sections 5.4 and 5.5 to describe and study the properties of an implementation technique for the λ-calculus. The underlying device, called the Categorical Abstract Machine (CAM), has been used to implement the first version of the programming language CAML (whose name was formed as a merge of "CAM" and "ML"). The last section presents an interesting blend of theory and practice, since at the same time a purely categorical theorem concerning free cartesian closed categories is proved.

5.1 Untyped λ-calculus

In this section, we present the λ-calculus and its basic computation rule – the β-reduction.

Definition 5.1.1 (λ-calculus) *The syntax of the untyped λ-calculus (λ-calculus for short) is given by:*

$$M ::= x \mid MM \mid \lambda x.M,$$

where x is called a variable, M_1M_2 is called an application, and $\lambda x.M$ is called an abstraction. The set of all λ-terms is denoted by Λ.

The following are frequently used abbreviations and terms:

$$\begin{array}{rcl} \lambda x_1 \cdots x_n.M & = & \lambda x_1.(\cdots \lambda x_n.M \cdots) \\ MN_1 \cdots N_n & = & (\cdots (MN_1) \cdots N_n) \end{array}$$

$$\begin{array}{ll} I = \lambda x.x & K = \lambda xy.x \\ \Delta = \lambda x.xx & S = \lambda xyz.(xz)(yz) \ . \end{array}$$

Definition 5.1.2 (head normal form) *A term* $\lambda x_1 \cdots x_n.xM_1 \cdots M_p$*, where* x *may or may not be equal to one of the* x_i*'s, is called a head normal form (hnf for short).*

Remark 5.1.3 *Any* λ*-term has exactly one of the following two forms: either it is a hnf, or it is of the form* $\lambda x_1 \cdots x_n.(\lambda x.M)M_1 \cdots M_p$ $(n \geq 0,\ p \geq 1)$.

Occurrences and contexts, which we introduce next, provide a notation allowing us to manipulate subterms.

Definition 5.1.4 (occurrence) *Let* M *be a term, and* u *be a word over the alphabet* $\{0,1,2\}$*. The subterm of* M *at occurrence* u*, written* M/u*, is defined as follows:*

$$\frac{}{M/\epsilon = M} \qquad \frac{M/u = N}{\lambda x.M/0u = N}$$

$$\frac{M_1/u = N}{M_1M_2/1u = N} \qquad \frac{M_2/u = N}{M_1M_2/2u = N}$$

where ϵ *is the empty word. The term* M/u *may well not be defined. If it is defined, we say that* u *is an occurrence of* M*. The result of replacing the subterm* M/u *by another term* N *is denoted* $M[u \leftarrow N]$.

Example 5.1.5 $(\lambda x.xy)/02 = y \quad (\lambda x.xy)[02 \leftarrow x] = \lambda x.xx$.

Definition 5.1.6 (context) *The contexts (with a unique hole) are defined by the following syntax (where* $i \in \omega$*):*

$$C ::= [\ ]_i \mid x \mid CM \mid MC \mid \lambda x.C\ .$$

where M *is a* λ*-term*

In Figure 5.1, we define the operation of filling the hole of a context by a term M. Of course, this just means replacing the hole by M, in the most naive way! Occurrences and contexts are related as follows.

Proposition 5.1.7 (occurrences/contexts) *For every term* M *and every occurrence* u *of* M*, there exists a unique context* C *such that* $M = C[M/u]$.

Free occurrences of variables are defined in Figure 5.2 through a predicate $\mathit{Free}(u, M)$. We define $\mathit{Bound}(u, v, M)$ (u is bound by v in M) by:

$$\frac{M/v = \lambda x.P \quad u = v0w \quad M/u = x \quad \mathit{Free}(w, P)}{\mathit{Bound}(u, v, M)}\ .$$

If we are not interested in the actual occurrences at which variables appear bound or free, we can define the sets $FV(M)$ and $BV(M)$ of free and bound variables of M by:

$$\begin{array}{rcl} FV(M) & = & \{x \mid \exists\, u\ \ M/u = x \text{ and } \mathit{Free}(u, M)\} \\ BV(M) & = & \{x \mid \exists\, u, v\ \ M/u = x \text{ and } \mathit{Bound}(u, v, M)\}\ . \end{array}$$

$$\begin{array}{lcl} [\,][N] & = & N \\ x[N] & = & x \\ (MC)[N] & = & M(C[N]) \\ (CM)[N] & = & (C[N]M \\ (\lambda x.C)[N] & = & \lambda x.(C[N]) \end{array}$$

Figure 5.1: Filling the holes of a context

$$\frac{}{\mathit{Free}(\epsilon, x)}$$

$$\frac{\mathit{Free}(u, M)}{\mathit{Free}(1u, MN)} \qquad \frac{\mathit{Free}(u, N)}{\mathit{Free}(2u, MN)} \qquad \frac{\mathit{Free}(u, M) \qquad M/u \neq x}{\mathit{Free}(0u, \lambda x.M)}$$

Figure 5.2: Free occurrences

If M is a term and $x \notin FV(M)$, one often says that x is fresh (relative to M).

The definition of substitution of a term for a (free) variable raises a difficulty. We expect $\lambda y.x$ and $\lambda z.x$ to be two different notations for the same thing: the constant function with value x. But careless substitution leads to:

$$(\lambda y.x)[x \leftarrow y] = \lambda y.y \qquad (\lambda z.x)[x \leftarrow y] = \lambda z.y\ ,$$

only the second of which is intended. What has gone wrong is that, in the first equality, the free variable y of the substituted term has been captured. This leads to the capture-avoiding definition of substitution given in Figure 5.3. The choice of z satisfying the side condition in the last clause of Figure 5.3 is irrelevant: we manipulate terms up to the following equivalence $\equiv$, called α-conversion:

$$(\alpha) \quad C[\lambda x.M] \equiv C[\lambda y.(M[x \leftarrow y])] \quad (y \notin FV(M))\ ,$$

for any occurrence context C and any term M.

We now introduce the basic computation rule of the λ-calculus.

Definition 5.1.8 (β-rule) *The β-rule is the following relation between λ-terms:*

$$(\beta) \quad C[(\lambda x.M)N] \ \rightarrow\ C[M[x \leftarrow N]]\ ,$$

$$\begin{array}{lcll} x[x \leftarrow N] & = & N & \\ y[x \leftarrow N] & = & y & (y \neq x) \\ (M_1M_2)[x \leftarrow N] & = & (M_1[x \leftarrow N])(M_2[x \leftarrow N]) & \\ (\lambda y.M)[x \leftarrow N] & = & \lambda z.(M[y \leftarrow z][x \leftarrow N]) & (z \notin FV(M) \cup FV(N)) \end{array}$$

Figure 5.3: Substitution in the λ-calculus

$$\overline{(\lambda x.M)N \ \rightarrow \ M[x \leftarrow N]}$$

$$(\nu)\ \frac{M \ \rightarrow \ M'}{MN \ \rightarrow \ M'N} \quad (\mu)\ \frac{N \ \rightarrow \ N'}{MN \ \rightarrow \ MN'} \quad (\xi)\ \frac{M \ \rightarrow \ M'}{\lambda x.M \ \rightarrow \ \lambda x.M'}$$

Figure 5.4: β-reduction

where C is an occurrence context and M, N are arbitrary terms. A term of the form $(\lambda x.M)N$ is called a redex.

In Figure 5.4, we give an alternative presentation of β-reduction, by means of an axiom and inference rules.

Exercise 5.1.9 *Show that the substitution algorithm as specified in Figure 5.3 is correctly defined, i.e., terminates.*

The λ-calculus, equipped with the β–reduction, is an example of a rewriting system. We shall use the following (standard) notation: we denote by $\rightarrow^*$ the reflexive and transitive closure of $\rightarrow$, and use $\rightarrow\ ^+$ to express that at least one step is performed. The reflexive, symmetric, and transitive closure of $\rightarrow$ is denoted simply with $=$.

Here are some examples of derivations, i.e., of sequences of successive rewritings.

Example 5.1.10 (1) $II \ \rightarrow \ I$.

(2) $SKK \rightarrow^* I$.

(3) $\Delta\Delta \ \rightarrow \ \Delta\Delta$.

(4) $(\lambda x.f(xx))(\lambda x.f(xx)) \ \rightarrow \ f((\lambda x.f(xx))(\lambda x.f(xx)))$.

The last two examples show that there are infinite reduction sequences. Moreover, the last example indicates how fixpoints can be encoded in the λ-calculus . If we set

$$Y = \lambda f.(\lambda x.f(xx))(\lambda x.f(xx)),$$

then we have $Yf =_\beta f(Yf)$. The term Y is known as Curry's fixpoint combinator.

Example 5.1.11 *Church numerals:*

$$\begin{aligned} &Succ = \lambda nfx.f(nfx) \\ &0 = \lambda fnx.x \end{aligned}$$

Here are two different ways to program addition, multiplication and exponential:

$$\begin{array}{lll} m+n & m\ Succ\ n & \lambda fx.m(f(nfx)) \\ m \times n & m\ (\lambda x.n+x)\ n & \lambda f.m(nf) \\ n^m & m\ (\lambda x.n \times x)\ n & mn \end{array}$$

Another rule, in addition to β, is often considered:

$$(\eta) \quad C[\lambda x.Mx] \;\rightarrow\; C[M] \quad (x \notin FV(M)) \; .$$

This is an extensionality rule, asserting that 'every term is a function' (if it is read backwards).

We state without proof the following result, which is one of the fundamental theorems of the λ-calculus.

Theorem 5.1.12 (Church-Rosser) *The β-reduction is confluent: If $M \rightarrow^* N$ and $M \rightarrow^* P$, then $N \rightarrow^* Q$ and $P \rightarrow^* Q$ for some Q.*

The following exercise (based on [16]) states a negative result due to Klop.

Exercise* 5.1.13 *Suppose that three constants D, F, S are added to the λ-calculus, together with the following new rewriting axiom:*

$$(SP) \quad D(Fx)(Sx) \;\rightarrow\; x \; .$$

Show that confluence fails for β+(SP). Hints: Consider the following so-called Turing fixpoint combinator:

$$Y_T = (\lambda xy.y(xxy))(\lambda xy.y(xxy)) \; .$$

The advantage of this term over Y (cf. example 5.1.10) is that $Y_T f$ is not only convertible to, but reduces to, $f(Y_T f)$. Set $C = Y_T(\lambda xy.D(F(Ey))(S(E(xy))))$ and $B = Y_T C$, where E is a free variable. Notice that $B \rightarrow^* A$ and $B \rightarrow^* CA$, where $A = E(CB)$. Show that A and CA have no common reduct, by contradiction, taking a common reduct with a minimum number of E's in head position.

5.2 A more precise look at substitution

In this section, we come back in more detail to the definition of substitution and of α-conversion, and provide a precise and rigorous treatment of these notions through de Bruijn notation (introduced below). This notation allows a "unique" representation of an α-equivalence class of terms. Note however that this unique choice is parameterized by a list of variables containing the free variables of the term. The treatment that follows allows us to separate substitution from α-conversion.

We begin by giving another version of the definition of substitution (due to Goubault [28]) which is now *partial*. It is based on the following definition of free and bound variables:

$$\begin{array}{ll}
FV(x) = \{x\} & BV(x) = \emptyset \\
FV(MN) = FV(M) \cup FV(N) & BV(MN) = BV(M) \cup BV(N) \\
FV(\lambda x.M) = FV(M) \setminus \{x\} & BV(\lambda x.M) = BV(M) \cup \{x\}
\end{array}$$

This definition coincides with that of the previous section for free variables, but not for the bound variables (for example, in the sense of the previous section we have $BV(\lambda x.y) = \emptyset$).

$$m[n \leftarrow N] = \begin{cases} m & \text{if } m < n \\ \tau_0^n(N) & \text{if } m = n \\ m-1 & \text{if } m > n \end{cases} \qquad \tau_i^n(j) = \begin{cases} j & \text{if } j < i \\ j+n & \text{if } j \geq i \end{cases}$$
$$(M_1M_2)[n \leftarrow N] = (M_1[n \leftarrow N])(M_2[n \leftarrow N]) \qquad \tau_i^n(N_1N_2) = (\tau_i^n(N_1))(\tau_i^n(N_2))$$
$$(\lambda.M)[n \leftarrow N] = \lambda.(M[n+1 \leftarrow N]) \qquad \tau_i^n(\lambda.N) = \lambda.(\tau_{i+1}^n(N))$$

Figure 5.5: Substitution in de Bruijn notation

Definition 5.2.1 *We say that x is substitutable by N in M if and only if $x \notin BV(M)$ and $FV(N) \cap BV(M) = \emptyset$.*

The point of this definition is that if these conditions hold, then the naive definition of the substitution is riskless:

$$\begin{array}{l} x[x \leftarrow N] = N \\ y[x \leftarrow N] = y \quad (y \neq x) \\ (M_1M_2)[x \leftarrow N] = M_1[x \leftarrow N]M_2[x \leftarrow N] \\ (\lambda y.M)[x \leftarrow N] = \lambda y.(M[x \leftarrow N]) \end{array}$$

In the rest of this section, we shall use this simplified definition.

We now move to de Bruijn notation. We introduce the following syntax:

$$M ::= n \mid MM \mid \lambda.M$$

where n is a natural number. The idea is to code a variable by its depth of binding. For the free variables, one searches this binding information in a list of variables that form somehow the environment of the term. The translation of the λ-calculus into the de Bruijn calculus is thus indexed by a list of variables (we write $|\vec{x}|$ for the length of $\vec{x}$, $x \in \vec{y}$ for the fact that x appears in the vector $\vec{y}$,...).

$$\begin{array}{l} DB_{\vec{x_1},x,\vec{x_2}}(x) = |\vec{x_2}| \quad (x \notin \vec{x_2}) \\ DB_{\vec{x}}(M_1M_2) = DB_{\vec{x}}(M_1)DB_{\vec{x}}(M_2) \\ DB_{\vec{x}}(\lambda y.M) = \lambda.DB_{\vec{x},y}(M) \end{array}$$

In de Bruijn notation, the substitution is defined as shown in Figure 5.5.

Here are some explanations. When N is substituted for (or reaches) a free occurrence of x in M, the binding depth of the bound variables of N is not modified, but that of the free variables needs to be adjusted (to avoid the capture of free variables). This explains the $\tau_0^n(N)$ and the two cases of the defintion of $\tau_i^n(j)$. Moreover, a λ disappears in the β-reduction, and this induces an adjustment of the free variables of $\lambda x.M$, whence the $m-1$ in the third case of the defintion of $m[n \leftarrow N]$. The crossings of λ are recorded by increments by 1.

The following exercises provide a justification for this rather involved definition.

Exercise 5.2.2 *For all $M, N, \vec{x_1}, x, \vec{x_2}$ such that $FV(M) \subseteq \vec{x_1}, x, \vec{x_2}$, $FV(N) \subseteq \vec{x_1}$, $x \notin \vec{x_2}$, $FV(N) \cap \vec{x_2} = \emptyset$ and x is substitutable by N in M, show:*

$$DB_{\vec{x_1},\vec{x_2}}(M[x \leftarrow N]) = DB_{\vec{x_1},x,\vec{x_2}}(M)[|\vec{x_2}| \leftarrow DB_{\vec{x_1}}(N)]$$

For all $N, \vec{x_1}, \vec{x_2}, \vec{x_3}$ such that $FV(N) \subseteq (\vec{x_1}, \vec{x_3})$ and $FV(N) \cap \vec{x_1} \cap \vec{x_2} \subseteq \vec{x_3}$, show:

$$\tau^{|\vec{x_2}|}_{|\vec{x_3}|}(DB_{\vec{x_1},\vec{x_3}}(N)) = DB_{\vec{x_1},\vec{x_2},\vec{x_3}}(N)$$

We define the axiom of β-reduction in de Bruijn notation as follows:

$$(\lambda.M)N \to_\beta M[0 \leftarrow N]$$

This is justified by the following proposition:

Proposition 5.2.3 *For all M, N and $\vec{x}$ such that $FV(M) \subseteq \vec{x}$ and $FV(N) \subseteq \vec{x}$, if $M \to_\beta N$, then $DB_{\vec{x}}(M) \to_\beta DB_{\vec{x}}(N)$.*

Proof. The case of the axiom is a instance of the first statement in Exercise 5.2.2, applied in the case where $\vec{x_2}$ is empty. □

In this section, α-equivalence $=_\alpha$ is redefined as the smallest congruence generated by:

$$\lambda x.M = \lambda y.M[x \leftarrow y] \quad (x \text{ substitutable by } y \text{ in } M \text{ and } y \notin FV(M))$$

Exercise 5.2.4

1. *For all M and all finite sets A of variables, show that there exists $M' =_\alpha M$ such that $BV(M') \cap A = \emptyset$.*
2. *For all M, x, N, show that there exists $M' =_\alpha M$ such that x is substitutable by N in M'.*

Exercise 5.2.5 *Let M, N be two λ-terms. Show that the following properties are equivalent:*

1. $M =_\alpha N$,
2. $\forall \vec{x}\ (FV(M) \cup FV(N) \subseteq \vec{x} \Rightarrow DB_{\vec{x}}(M) = DB_{\vec{x}}(N))$,
3. $\exists \vec{x}\ (FV(M) \cup FV(N) \subseteq \vec{x}$ *and* $DB_{\vec{x}}(M) = DB_{\vec{x}}(N))$.

Finally, we redefine the β-reduction as follows, on α-equivalence classes:

$$\frac{N \text{ is substitutable to } x \text{ in } M}{[(\lambda x.M)] \to [M[x \leftarrow N]]} \quad \frac{[M] \to [N]}{[\lambda x.M] \to [\lambda x.N]} \quad \frac{[M_1] \to [M_1']}{[M_1 M_2] \to [M_1' M_2]} \quad \frac{[M_2] \to [M_2']}{[M_1 M_2] \to [M_1 M_2']}$$

Proposition 5.2.6 *The β-reduction is correctly defined, i.e., if $(\lambda x.M)N =_\alpha (\lambda x'.M')N'$, if x is substitutable by N in M and if x' is substitutable by N' in M', then*

$$M[x \leftarrow N] =_\alpha M'[x' \leftarrow N'] \; .$$

PROOF. Let $\vec{z}$ such that $FV((\lambda x.M)N) \subseteq \vec{z}$. Then

$$DB_{\vec{z}}((\lambda x.M)N) = DB_{\vec{z}}((\lambda x'.M')N') \quad \text{(by Exercise 5.2.5)},$$

hence $DB_{\vec{z}}(N) = DB_{\vec{z}}(N')$ and $DB_{\vec{z},x}(M) = DB_{\vec{z},x'}(M')$. We obtain

$$\begin{array}{rcl} DB_{\vec{z}}(M[x \leftarrow N]) & = & DB_{\vec{z},x}(M)[0 \leftarrow DB_{\vec{z}}(N)] \\ & = & DB_{\vec{z},x'}(M')[0 \leftarrow DB_{\vec{z}}(N')] \quad = \quad DB_{\vec{z}}(M'[x' \leftarrow N']) \end{array}$$

by a double application of Exercise 5.2.2. We finally conclude by applying again Exercise 5.2.5. □

Exercise 5.2.7 *Show that the definitions of α–conversion and of substitution given in the previous section are correct, i.e., two terms are α–convertible in the sense of this section if and only if and only if they are so in the sense of the previous section, and for all M, x, N there exists M' such that $M' =_\alpha M$, x is substitutable by N in M' and $M[x \leftarrow N] =_\alpha M'[x \leftarrow N]$, where the substitution on the left is that of the preceding section and the one on the right is that of this section. (Indication: go through de Bruijn notation).*

5.3 Categorical combinators

In this section, we relate λ-calculus and (cartesiant closed) categories, by means of a syntactic translation from λ-calculus (in De Bruijn notation) to CCC's, viewed as a syntax. On the way, we introduce the simply typed λ-calculus. We describe an implementation of the λ-calculus, based on this translation.

We provide a syntax of types:

$$A ::= \texttt{int} \mid A \rightarrow A$$

(`int` is a basic type, there could be other such basic types). The simply typed terms are the λ-terms accepted by the following typing system, where Γ is a list of variable type declarations of the form $x : A$:

$$\frac{x \text{ does not appear in } \Delta}{\Gamma, x : A, \Delta \vdash x : A} \qquad \frac{\Gamma \vdash M : A \rightarrow B \quad \Gamma \vdash N : A}{\Gamma \vdash MN : B} \qquad \frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x.M : A \rightarrow B}$$

The simply typed λ-calculus in de Bruijn notation is defined via the following typing rules (where the contexts are now lists of types):

$$\frac{i \leq n}{A_n, \ldots, A_0 \vdash i : A_i} \qquad \frac{\Gamma \vdash M : A \rightarrow B \quad \Gamma \vdash N : A}{\Gamma \vdash MN : B} \qquad \frac{\Gamma, A \vdash M : B}{\Gamma \vdash \lambda.M : A \rightarrow B}$$

We now move to cartesian closed categories, which we present here in a purely syntactic (and self-contained) way. The language of types is enriched with finite products (1 standing for the empty product):

$$A ::= \texttt{int} \mid 1 \mid A \times A \mid A \rightarrow A$$

The (untyped) terms, also called categorical combinators, are given by the following syntax:

$$f ::= id \mid f \circ f \mid \langle f, g \rangle \mid \pi \mid \pi' \mid \Lambda(f) \mid ev \mid !$$

The typed terms are the terms accepted by the following typing system, where the judgements have the form $A \vdash f : B$ (cf. Chapter 1).

$$\frac{}{A \vdash id : A} \qquad \frac{A \vdash f : B \quad B \vdash g : C}{A \vdash g \circ f : C} \qquad \frac{}{A \vdash ! : 1}$$

$$\frac{A \vdash f : B \quad A \vdash g : C}{A \vdash \langle f, g \rangle : B \times C} \qquad \frac{}{A \times B \vdash \pi : A} \qquad \frac{}{A \times B \vdash \pi' : B}$$

$$\frac{A \times B \vdash f : C}{A \vdash \Lambda(f) : B \rightarrow C} \qquad \frac{}{(A \rightarrow B) \times A \vdash ev : B}$$

We consider categorical combinators up to provable equality with respect to the following set of equations:

$$\begin{array}{l}
(f \circ g) \circ h = f \circ (g \circ h) \\
f \circ id = f \\
id \circ f = f \\
\\
\pi \circ \langle f, g \rangle = f \\
\pi' \circ \langle f, g \rangle = g \\
\langle f, g \rangle \circ h = \langle f \circ h, g \circ h \rangle \\
\langle \pi, \pi' \rangle = id \\
\\
ev \circ \langle \Lambda(f), g \rangle = f \circ \langle id, g \rangle \\
\Lambda(f) \circ g = \Lambda(f \circ \langle g \circ \pi, \pi' \rangle) \\
\Lambda(ev) = id \\
\\
f = !
\end{array}$$

All the equations (except the last one) can be considered without reference to types, i.e., define also a calculus or untyped categorical combinators. The equation $f =!$ makes sense only for $A \vdash f : 1$. We shall forget it temporarily.

We next shall compile the λ-calculus in de Bruijn notation into the syntax of categorical combinators, as follows. Types are translated into themselves. As for (typed) terms, a judgement of type $\Gamma \vdash M : A$ is translated into a morphism

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

We first need to interpret contexts:

$$[\![\ ]\!] = 1 \qquad [\![\Gamma, A]\!] = [\![\Gamma]\!] \times A$$

Il is instructive to represent this translation in a tree form, as it will then be straightforward to read the interpretation of variables beneath as an access path in a tree. The translation of the typing judgements is the following:

$$\begin{array}{lcl} [\![A_n, \ldots, A_0 \vdash i : A_i]\!] & = & (\ldots(1 \times A_n) \times \ldots) \times A_0 \vdash \pi' \circ \pi^i : A_i \\ [\![\Gamma \vdash MN : B]\!] & = & ev\circ < [\![\Gamma \vdash M : A \to B]\!], [\![\Gamma \vdash N : A]\!] > \\ [\![\vdash \lambda.M : A \to B]\!] & = & \Lambda([\![\Gamma, A \vdash M : B]\!]) \end{array}$$

Even if this translation is easier to understand in a typed framework, it can be as well and more soberly expressed in an untyped framework:

$$[\![i]\!] = \pi' \circ \pi^i \qquad [\![MN]\!] = ev\circ < [\![M]\!], [\![N]\!] > \qquad [\![\lambda.M]\!] = \Lambda([\![M]\!])$$

We shall now show how the categorical equations (oriented from left to right) allow us to simulate the β-reduction. A β-reduction step will be simulated by several categorical rewriting steps. This is due to the fact that in the λ-calculus, the substitution is defined outside the calculus itself, while the substitution process is gradual in categories: in a substitution $M[x \leftarrow N]$, N traverses M progressively down to the leaves x. In order to fill this gap, we extend the syntax of the λ-calculus of de Bruijn with operators that render explicit, or internalise, substitution, and with corresponding rules (oriented from left to right):

$$M ::= n \mid MM \mid \lambda.M \mid M[n \leftarrow N] \mid \tau_i^n(N)$$

The following exercise should make it easier to understand the details of the definition of substitution in de Bruijn notation.

Exercise 5.3.1 *Write typing rules for $M[n \leftarrow N]$ and $\tau_i^n(N)$.*

We complete the categorical interpretation as follows:

$$[\![M[n \leftarrow N]]\!] = [\![M]\!] \circ P^n(\langle id, [\![N]\!]\rangle) \qquad [\![\tau_i^n(N)]\!] = [\![N]\!] \circ P^i(\pi^n)$$

where $P(f) = f \times id = \langle f \circ \pi, \pi' \rangle$ (and hence $\Lambda(f) \circ g = \Lambda(f \circ P(g))$). The powers of the operator P correspond to the successive traversals of λ nodes. The simulation begins as follows (we omit the symbols $[\![$ and $]\!]$).

$$(\lambda.M)N \to M[0 \leftarrow N] \quad \text{reads as} \quad ev \circ \langle \Lambda(M), N \rangle \to M \circ \langle id, N \rangle$$

This allows us to assert a first slogan:

$$\textit{substitution} = \textit{composition}$$

The defintion of $[\![M[n \leftarrow N]]\!]$ for $n > 0$ (as well as that of $[\![\tau_i^n(N)]\!]$) is "dictated" by the traversal of λ nodes. The rule $(\lambda.M)[n \leftarrow N] = \lambda.(M[n+1 \leftarrow N])$ (resp. $\tau_i^n(\lambda.N) = \lambda.(\tau_{i+1}^n(N)))$ is then simulated using the rule

$$\Lambda(f) \circ g \rightarrow \Lambda(f \circ \langle g \circ \pi, \pi' \rangle)$$

The traversal of applications is simulated thanks to the associativity rule and to the rule

$$\langle f, g \rangle \circ h \rightarrow \langle f \circ h, g \circ h \rangle$$

This allows us to assert a second slogan:

$$\text{passage of the substitution} = \text{natural transformation}$$

We recommend to the reader to check that these two categorical equations indeed express the naturality in A and C of the bijections between $\mathbf{C}[A, B] \times \mathbf{C}[A, C]$ and $\mathbf{C}[A, B \times C]$ and between $\mathbf{C}[C \times A, B]$ and $\mathbf{C}[C, A \rightarrow B]$, respectively.

We are left with the variable case:

$$[\![m[n \leftarrow N]]\!] = (\pi' \circ \pi^m) \circ P^n(< id, [\![N]\!] >)$$

We note that by definition of P, and thanks to the two projection rules and to the right identity rule, we have:

$$\pi \circ P(f) \rightarrow f \circ \pi \qquad \pi' \circ P(f) \rightarrow^* \pi'$$

We examine the three cases (cf. Figure 5.5):

- $m < n$. Repeating the commutation of π and P, we obtain:

  $$\pi' \circ \pi^m \circ P^n(\langle id, [\![N]\!] \rangle) \rightarrow^* \pi' \circ P^{n-m}(\langle id, N \rangle) \circ \pi^m \rightarrow^* \pi^m$$

  (proceeding with care, one may use the associativity rule only from left to right).

- $m = n$. We have in this case:

  $$\pi' \circ \pi^n \circ P^n(\langle id, [\![N]\!] \rangle) \rightarrow^* \pi' \circ \langle id, N \rangle \circ \pi^n \rightarrow^* [\![N]\!] \circ \pi^m$$

  whence the defintion of $[\![\tau_0^n(N)]\!]$.

- $m > n$. We have:

  $$\pi' \circ \pi^m \circ P^n(\langle id, [\![N]\!] \rangle) \rightarrow^* \pi' \circ \pi^{m-n} \circ \langle id, [\![N]\!] \rangle \circ \pi^n \rightarrow^* \pi' \circ \pi^{m-1}$$

Finally, we examine $[\![\tau_i^n(j)]\!]$.

- $j < i$. We have:

$$\pi' \circ \pi^j \circ P^i(\pi^n) \to^* \pi' \circ P^{i-j}(\pi^n) \circ \pi^j \to^* \pi' \circ \pi^j$$

- $j \geq i$.

$$\pi' \circ \pi^j \circ P^i(\pi^n) \to^* \pi' \circ \pi^{j-i} \circ \pi^n \circ \pi^i \to^* \pi' \circ \pi^{j+n}$$

which completes the proof of correctness of the simulation.

Remark 5.3.2 *1. We did not use the equation $\Lambda(ev) = id$. It allows us to validate the rule of η-conversion $\lambda x.Mx = M$.*

2. We could have extended the λ-calculus to product types:

$$M ::= x \mid MM \mid \lambda x.M \mid fst(M) \mid snd(M) \mid (M, M) \mid !$$

The simulation of the β-reduction extends without difficulty. The rule $\langle \pi, \pi' \rangle = id$ allows us to validate the "η-conversion of the product", or surjective pairing, *which is the following rule:*

$$(fst(M), snd(M)) = M$$

In fact, there is a complete equivalence between λ-calculus with products and categorical combinators.

3. Categorical combinators [15] have been the first calculus to provide an explicit computational treatment of substitution. Since then, syntaxes closer to that of the λ-calculus have been proposed, the first one being the $\lambda\sigma$-calculus [1, 17].

Exercise 5.3.3 *Justify the claims about η and about surjective pairing made in Remark 5.3.2.*

5.4 The Categorical Abstract Machine

In this section, we show how to execute categorical combinators, not by rewriting, but by defining an evaluation function, by inference rules, From the specification of the evaluation function, we shall in turn extract an abstract machine expressed by a transition system.

The idea underlying the definition of the evaluation function is to formalise the set-theoretical functions underlying the combinators. In the following rules, the notation $\langle f|s \rangle = t$ should read as: "the value of f at s is t.

$$\frac{}{\langle id|s\rangle = s} \qquad \frac{\langle f|s\rangle = s_1 \;\; \langle g|s_1\rangle = s_2}{\langle g\circ f|s\rangle = s_2}$$

$$\frac{\langle f|s\rangle = s_1 \;\; \langle g|s\rangle = s_2}{\langle\langle f,g\rangle|s\rangle = (s_1,s_2)} \qquad \frac{}{\langle \pi|(s_1,s_2)\rangle = s_1} \qquad \frac{}{\langle \pi'|(s_1,s_2)\rangle = s_2}$$

$$\frac{}{\langle \Lambda(f)|s\rangle = \Lambda(f)s} \qquad \frac{\langle f|(s,t)\rangle = s_1}{\langle ev|(\Lambda(f)s,t)\rangle = s_1}$$

On the way, we have defined a syntax for new syntactical objects s, that we call *values*:

$$s ::= () \mid \underline{n} \mid (s,s) \mid \Lambda(f)s$$

The value () is used for evaluating a closed term, while $\underline{n}$ is a constant of basic type. For example:

$$\langle\Lambda(\pi')|()\rangle = \Lambda(\pi')() \quad \langle ev\circ < \Lambda(\pi'),\pi' > |\underline{2}\rangle = \underline{2}$$

The construction $\Lambda(f)s$ is inspired from the practice of functional programming languages implementation: it is a *closure* that stores the code of a function together with its environment at declaration time. In a functional langage like CAML or Haskell, evaluation is *weak*, i.e., does not compute under λ's: for example, $\lambda x.(\lambda y.y)x$ is not evaluated, but $(\lambda x.(\lambda y.y)x)\underline{2}$ is (yielding value $\underline{2}$). The passage of $\lambda x.(\lambda y.y)x$ to $\lambda x.x$ is a program transformation, or optimisation, which can be taken care of by different tools, such as partial evaluation.

The above formal system is deterministic, since there is exactly one rule per combinator. We shall show that it is total, in the sense that $\langle f|s\rangle$ is always well defined (if f and s are correctly typed, and if their types match).

But we first derive an abstract machine from our specification of the evaluation function. Since this specification is recursive terminal, it is straightforward to turn it into an iterative one, by means of a stack. We now view a term as a piece of code: composition becomes code concatenation, and the symbols of the pairing combinator become instructions `PUSH`, `SWAP` and `CONS`, respectively. Here are the transitions of the machine:

$$\begin{array}{l}
code(id) = \mathtt{SKIP} \quad code(\pi) = \mathtt{CAR} \quad code(\pi') = \mathtt{CDR} \quad code(ev) = ev \\
code(g\circ f) = code(f); code(g) \quad code(\Lambda(f)) = \Lambda(code(f)) \\
code(\langle f,g\rangle) = \mathtt{PUSH}; code(f); \mathtt{SWAP}; code(g); \mathtt{CONS}
\end{array}$$

$$
\begin{array}{lcl}
\langle \mathtt{SKIP}; C \mid s \mid S\rangle & \to & \langle C \mid s \mid S\rangle \\
\langle \mathtt{CAR}; C \mid (s_1, s_2) \mid S\rangle & \to & \langle C \mid s_1 \mid S\rangle \\
\langle \mathtt{CDR}; C \mid (s_1, s_2) \mid S\rangle & \to & \langle C \mid s_2 \mid S\rangle \\
\langle \mathtt{PUSH}; C \mid s \mid S\rangle & \to & \langle C \mid s \mid s \cdot S\rangle \\
\langle \mathtt{SWAP}; C \mid s_1 \mid s_2 \cdot S\rangle & \to & \langle C \mid s_2 \mid s_1 \cdot S\rangle \\
\langle \mathtt{CONS}; C \mid s_2 \mid s_1 \cdot S\rangle & \to & \langle C \mid (s_1, s_2) \mid S\rangle \\
\langle \Lambda(C); C' \mid s \mid S\rangle & \to & \langle C' \mid \Lambda(C)s \mid S\rangle \\
\langle ev; C' \mid (\Lambda(C)s, t) \mid S\rangle & \to & \langle C; C' \mid (s,t) \mid S\rangle
\end{array}
$$

This machine is called the CAM (Categorical Abstract Machine) [14]. The three components are called code, environment and stack, respectively. The two evaluation mechanisms are related as follows.

Proposition 5.4.1 *If $\langle f|s\rangle = t$, then $\langle f \mid s \mid\, \rangle \to^* \langle\, \mid t \mid\, \rangle$ (the empty space means empty stack or empty code).*

Proof. Left to the reader.

5.5 Termination of the CAM and hierarchical coherence

In this section, we show that the (typed) CAM terminates, i.e. does not admit infinite sequences of successive transitions, and we prove an invariant of the evaluation fonction, a consequence of which is the following theorem, due to Lafont [33].

Theorem 5.5.1 *The embedding of a category $\mathbf{C}$ in the free cartersian closed category $CCC(\mathbf{C})$ generated by $\mathbf{C}$ is full and faithful.*

We have to accommodate the datas of the category $\mathbf{C}$. To this aim, we adjust the syntax of types and categorical combinators:

$$
\begin{array}{l}
A ::= \underline{A} \mid 1 \mid A \times A \mid A \to A \\
f ::= \underline{f} \mid id \mid f \circ f \mid \langle f, g\rangle \mid \pi \mid \pi' \mid \Lambda(f) \mid ev \mid !
\end{array}
$$

The clauses $\underline{A}$ and $\underline{f}$ import the objets and the morphisms of $\mathbf{C}$.

We add the following typing judgement:

$$
\frac{\underline{f} \in \mathbf{C}[\underline{A}, \underline{B}]}{\underline{A} \vdash \underline{f} : \underline{B}}
$$

The free cartersian closed category $CCC(\mathbf{C})$ is the category whose objets are the types A generated by the syntax of types, and the morphisms are the categorical terms that are generated by the syntax of terms, quotiented by the equations given above, together with

$$
\begin{array}{l}
\underline{id} = id \\
\underline{f \circ g} = \underline{f} \circ \underline{g}
\end{array}
$$

We also adjust the syntax of values:

$$s ::= \underline{f} \mid (s,s) \mid \Lambda(f)s \mid !$$

We shall type values by means of judgements of the form $\vdash_{\underline{A}} s : B$, and at the same time we shall define a "decompilation" $s \mapsto {}'s$ from values to terms (such that $\underline{A} \vdash {}'s : B$):

$$\frac{\underline{f} \in \mathbf{C}[\underline{A},\underline{B}]}{\vdash_{\underline{A}} \underline{f} : \underline{B} \qquad {}'\underline{f} = \underline{f}} \qquad\qquad \frac{}{\vdash_{\underline{A}} ! : 1 \qquad {}'! = !}$$

$$\frac{\vdash_{\underline{A}} s : B \quad \vdash_{\underline{A}} t : C}{\vdash_{\underline{A}} (s,t) : B \times C \qquad {}'(s,t) = \langle {}'s, {}'t \rangle}$$

$$\frac{C \times B_1 \vdash f : B_2 \quad \vdash_{\underline{A}} s : C}{\vdash_{\underline{A}} \Lambda(f)s : B_1 \to B_2 \qquad {}'(\Lambda(f)s) = \Lambda(f) \circ {}'s}$$

Note that the transformation $'$ is essentielly an inclusion from values to terms: we could as well have written $\langle s,t \rangle$ instead of (s,t) and $\Lambda(f) \circ s$ instead of $\Lambda(f)s$ in the first place, but the notation change underlines the fact that this subset of terms plays a distinct role.

We extend the definition of the evaluator by the following clauses:

$$\frac{}{\langle \underline{f} | \underline{g} \rangle = \underline{f \circ g}} \qquad\qquad \frac{}{\langle ! | s \rangle = !}$$

The key property is stated in the following proposition.

Proposition 5.5.2 *If $B \vdash f : C$ and $\vdash_{\underline{A}} s : B$, then there exists a (unique) t such that $\langle f | s \rangle = t$. Moreover, we have $\vdash_{\underline{A}} t : C$.*

Proof. We use a *realisability* technique (in a particularly simple form) . We write $\langle f | s \rangle \downarrow$ if there exists t such that $\langle f | s \rangle = t$. We define the predicate $s \Vdash_{\underline{A}} B$ (s realises B) as follows:

$$\frac{\vdash_{\underline{A}} \underline{f} : B}{\underline{f} \Vdash_{\underline{A}} B} \qquad \frac{s \Vdash_{\underline{A}} B \quad t \Vdash_{\underline{A}} C}{(s,t) \Vdash_{\underline{A}} B \times C} \qquad \frac{\forall t \ \ (t \Vdash_{\underline{A}} B_1 \Rightarrow \langle f | (s,t) \rangle \downarrow)}{\Lambda(f)s \Vdash_{\underline{A}} B_1 \to B_2}$$

The statement is an immediate consequence of the following properties:

(A) If $B \vdash f : C$ and $s \Vdash_{\underline{A}} B$, then there exists t such that $\langle f | s \rangle = t$, and moreover $t \Vdash_{\underline{A}} C$. This is proved by induction on f. In the case of the composition, we use the clause "moreover $t \Vdash_{\underline{A}} C$". The definition of $\Lambda(f)s \Vdash_{\underline{A}} B_1 \to B_2$ is precisely designed to pass the case *ev*. All the other cases are evident.

(B) If $\vdash_{\underline{A}} s : B$, then $s \Vdash_{\underline{A}} B$. We proceed here by induction on s. The only non trivial case is $\Lambda(f)s$. By induction, we have $s \Vdash_{\underline{A}} C$. Let t be such that $t \Vdash_{\underline{A}} B_1$. We have $(s,t) \Vdash_{\underline{A}} C \times B_1$, hence by property (A) (which we have proved independently), we have $\langle f|(s,t)\rangle \downarrow$. Hence $\Lambda(f)s \Vdash_{\underline{A}} B_1 \to B_2$. This concludes the proof. □

Corollary 5.5.3 *The CAM (executed on correctly typed terms) terminates.*

Proof. Cf. Proposition 5.4.1. □

But there is more to this story. We now prove a *coherence* result.

Proposition 5.5.4

$$(B \vdash f : C\ ,\ \vdash_{\underline{A}} s : B\ ,\ \langle f|s\rangle = t) \quad \Rightarrow \quad \vdash f \circ {}'s = {}'t$$

Proof. The proof is by induction on the proof that $\langle f|s\rangle = t$. We use the same equations as for the simulation of β, with two differences:

1. we do not use the naturaliy of Λ (weak evaluation!);
2. we use a variant of the rule that allows us to initiate the β-reduction:

$$ev \circ \langle \Lambda(f) \circ h, g\rangle = f \circ \langle h, g\rangle$$

.

We can now prove the hierarchical coherence theorem

Proof of Theorem 5.5.1.

- The embedding is full. Let $\underline{A} \vdash f : \underline{B}$, and let $s = \langle f|\underline{id}\rangle$ (with $\vdash_{\underline{A}} id : \underline{A}$). (We now that s exists by totality.) By the typing, we know that $s = \underline{g}$, for some g, with $\vdash_{\underline{A}} \underline{g} : \underline{B}$. By coherence, we have $\vdash f \circ {}'id = {}'\underline{g}$, hence $\vdash f = \underline{g}$.
- The embedding is faithful. We can show that for all categorical axioms $f = g$, we have $\langle f|s\rangle = \langle g|s\rangle$ for all (correctly typed) s. Thus if $\vdash \underline{f} = \underline{g}$, we have (using the clause of the evaluator for the constants $\underline{f}$, $\underline{g}$):

$$\underline{f} = \underline{f \circ id} = \langle \underline{f}|\underline{id}\rangle = \langle \underline{g}|\underline{id}\rangle = \underline{g \circ id} = \underline{g}$$

i.e., if $\underline{f}$ and $\underline{g}$ are equal modulo the equations that characterise $CCC(\mathbf{C})$, then $\underline{f}$ and $\underline{g}$ *coincide.* □

Note that Theorem 5.5.1 holds as well for a slightly weaker quotient (by the equational theory described just before the proof of the hierarchical coherence property.

Bibliography

[1] M. Abadi, L. Cardelli, P.-L. Curien, and J.-J. Lévy, Explicit Substitutions, Journal of Functional Programming 1(4), 375-416 (1992).

[2] R. Amadio, and Pierre-Louis Curien, Domains and Lambda-calculi, Cambridge University Press (1998).

[3] Barendregt, H., The lambda-calculus ; its syntax and semantics, North-Holland, (1984).

[4] M. Barr and C. Wells, Toposes, triples and theories, Springer-Verlag (1985).

[5] J. Beck, Distributive laws, Lecture Notes in Mathematics 80, 119-140 (1969).

[6] J. Bénabou, Introduction to bicategories, Lecture Notes in Mathematics 40, 1-77 (1967).

[7] J. Bénabou, Les distributeurs, Université Catholique de Louvain, Institut de Mathématique Pure et Appliquée, Rapport 33 (1973).

[8] F. Borceux, Handbook of categorical algebra, vol. I: basic category theory, Cambridge University Press (1994).

[9] F. Borceux, Handbook of categorical algebra, vol. II: categories and structures, Cambridge University Press (1994).

[10] N. de Bruijn, Lambda-calculus notation with nameless dummies, a tool for automatic formula manipulation, Indag. Math. 34, 381-392 (1972).

[11] A. Burroni, T-catégories (catégories dans un triple), Cahiers de Topologie et Géométrie Différentielle XII(3), 215-321 (1971).

[12] A. Burroni, Higher Dimensional Word Problem, Theoretical Computer Science 115, 43-62 (1993).

[13] Church, A., The calculi of lambda-conversion, Princeton University Press (1941).

[14] G. Cousineau, P.-L. Curien, and M. Mauny, The categorical abstract machine, Sci. of Comp. Programming 8, 173-202 (1987).

[15] P.-L. Curien, Categorical combinators, sequential algorithms and functional programming, Pitman (1986), revised edition, Birkhäuser (1993).

[16] P.-L. Curien and T. Hardin, Yet yet a counterexample for λ+SP, Journal of Functional Programming 4(1), 113-115, (1994).

[17] P.-L. Curien, T. Hardin, and J.-J. Lévy, Weak and Strong Confluent Calculi of Explicit Substitutions, Journal of the ACM 43(2) (1996).

[18] P.-L. Curien, Introduction to linear logic and ludics, part I, Advances in Mathematics (China) 34 (5), 513-544 (2005).

[19] P.-L. Curien, Introduction to linear logic and ludics, part II, Advances in Mathematics (China) 35 (1), 1-44 (2006).

[20] Curry, H. and Feys, R., Combinatory Logic, vol. 1, North Holland, (1958).

[21] Curry, H. and Hindley, R. and Seldin, J., Combinatory Logic, vol. 2, North Holland (1972).

[22] B.J. Day, On closed categories of functors, Reports of the Midwest Category Seminar IV, Lecture Notes in Mathematics 137, 1-38 (1970).

[23] M. Fiore, Mathematical models of computational and combinatorial structures. Invited address for Foundations of Software Science and Computation Structures (FOSSACS 2005), Lecture Notes in Computer Science 3441, 25-46 (2005).

[24] M. Fiore, N.Gambino, M.Hyland, and G.Winskel, The cartesian closed bicategory of generalised species of structures. Preprint (2006).

[25] N. Gambino, On the coherence conditions for pseudo-distributive laws, draft.

[26] R. Garner, Polycategories, PhD Thesis, Cambridge University (2005).

[27] G.B. Im and G.M. Kelly, A universal property of the convolution monoidal structure, Journal of Pure and Applied Algebra 43, 75-88 (1986).

[28] J. Goubault-Larrecq, Lambda-calcul et langages fonctionnels, Notes de cours, available from `http://www.lsv.ens-cachan.fr/~goubault`.

[29] Hindley, R. and Seldin, J., Introduction to combinators and lambda-calculus , Cambridge University Press (1986).

[30] A. Joyal, Foncteurs analytiques et espèces de structures, in Combinatoire énumérative, Lecture Notes in Mathematics 1234, 126-159 (1986).

[31] M. Kelly, On the operads of J.P. May, Reprints in Theory and Applications of Categories 13, 1-13 (2005).

[32] J.-L. Krivine, Lambda-calculus, types et modèles, Masson (1991).

[33] Y. Lafont, Logiques, catégories et machines, Thèse de doctorat, Université Paris 7 (1988).

[34] Y. Lafont, Towards an algebraic theory of Boolean circuits, Journal of Pure and Applied Algebra 184 (2-3), 257-310, Elsevier (2003).

[35] T. Leinster, Higher operads, higher categories, London Mathematical Society Lecture Note Series 298, Cambridge University Press (2004).

[36] S. Mac Lane, Categories for the working mathematician, Springer-Verlag (1971).

[37] M. Tanaka and J. Power, Pseudo-distributive laws and axiomatics for variable binding, Higher-Order and Symbolic Computation 19(2-3), 305 - 337 (2006).

[38] B. Vallette, Dualité de Koszul des PROPS, Thèse de Doctorat, Université de Strasbourg (2003).

[39] B. Vallette, A Koszul duality for props, to appear in Transactions of A.M.S.