On my favorite conventions for drawing the missing diagrams in Category Theory

Eduardo Ochs

July 24, 2021

Abstract

I used to believe that my conventions for drawing diagrams for categorical statements could be written down in one page or less, and that the only tricky part was the technique for reconstructing objects "from their names" (sec.3)... but then I found out that this is not so.

This is an attempt to explain, with motivations and examples, all the conventions behind a certain diagram, called the "Basic Example" in the text. Once the conventions are understood that diagram becomes a "skeleton" for a certain lemma related to the Yoneda Lemma, in the sense that both the statement and the proof of that lemma can be reconstructed from the diagram. The last sections discuss some simple ways to extend the conventions; we see how to express in diagrams the ("real") Yoneda Lemma and a corollary of it, how to define comma categories, and how to formalize the diagram for "geometric morphism for children" mentioned in sec.1.

People in CT usually only share their ways of visualizing things when their diagrams cross some threshold of mathematical relevance — and this usually happens when they prove new theorems with their diagrams, or when they can show that their diagrams can translate calculations that used to be huge into things that are much easier to visualize. The diagrammatic language that I present here lies below that threshold — and so it is a "private" diagrammatic language, that I am making public as an attempt to establish a dialogue with other people who have also created their own private diagrammatic languages.

Contents

1 Missing diagrams

I need to tell a long story here.

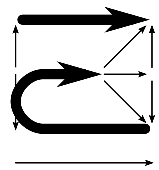
Let me start with some quotes. This one is from Eilenberg and Steenrod ([ES52, p.ix], but I learned it from [Krö07, pp.82–83]):

> The diagrams incorporate a large amount of information. Their use provides extensive savings in space and in mental effort. In the case of many theorems, the setting up of the correct diagram is the major part of the proof. We therefore urge that the reader stop at the end of each theorem and attempt to construct for himself the relevant diagram before examining the one which is given in the text. Once this is done, the subsequent demonstration can be followed more readily; in fact, the reader can usually supply it himself.

I spent a *lot* of my time studying Category Theory trying to "supply the diagrams myself". In [ES52] supplying the diagrams is not very hard (I guess), but in books like [CWM], in which most important concepts involve several categories, I had to rearrange my diagrams hundreds of times until I reached "good" diagrams...

The problem is that I expected too much from "good" diagrams. The next quotes are from the sections 1 and 12 of an article that I wrote about that ([IDARCT]):

> My memory is limited, and not very dependable: I often have to rededuce results to be sure of them, and I have to make them fit in as little "mental space" as possible...
>
> Different people have different measures for "mental space"; someone with a good algebraic memory may feel that an expression like $\mathsf{Frob} : \Sigma_f(P \land f^*Q) \cong \Sigma_f P \land Q$ is easy to remember, while I always think diagramatically, and so what I do is that I remember this diagram,

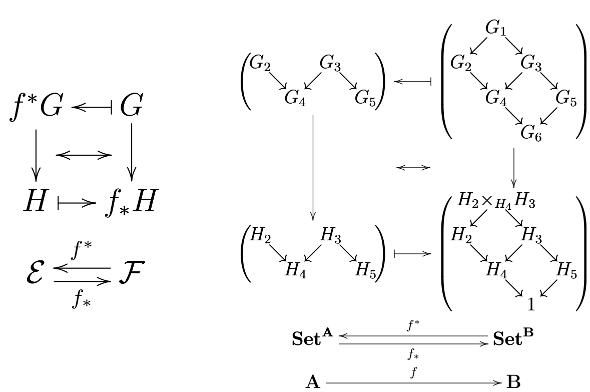


and I reconstruct the formula from it.

> Let's call the "projected" version of a mathematical object its "skeleton". The underlying idea in this paper is that for the right kinds of projections, and for some kinds of mathetical objects, it should be possible to reconstruct enough of the original object from its skeleton and few extra clues — just like paleontologists can reconstruct from a fossil skeleton the look of an animal when it was alive.

I was searching for a diagrammatic language that would let me express the "skeletons" of categorical definitions and proofs. I wanted these skeletons to be easy to remember — partly because they would have shapes that were easy to remember, and partly because they would be similar to "archetypal cases" ([IDARCT, section 16]).

In 2016 and 2017 I taught a seminar course for undergraduates that covered a bit of Category Theory in the end — see Section 5.5 and [Och19] — and this forced me to invent new techniques for working in two different styles in parallel: a style "for adults", more general, abstract, and formal, and another "for children", with more diagrams and examples. After some semesters, and after writing most of the material that became [PH1], I tried to read again some parts of Johnstone's "Sketches of an Elephant", a book that always felt quite impenetrable to me, and I found a way to present geometric morphisms in toposes to "children". It was based on this diagram,



that we will discuss in detail in 7.13. Its left half is a generic geometric morphism ("for adults"), and its right half is a very specific geometric morphism ("for children") in which everything is easy to understand and to visualize, and that turns out to be "archetypal enough".

I showed that to the few categorists with whom I had contact and the feedback that I got was quite positive. A few of them — the ones who were strictly "adults" — couldn't understand why I was playing with particular cases, and even worse, with finite categories, instead of proving things in the most general case possible, but some others said that these ideas were very nice, that they knew a few bits about geometric morphisms but those bits didn't connect well, and that now they had a family of particular cases to think about, and they had much more intuition than before.

That was the first time that my way of using diagrams yielded something so nice! This was the excuse that I needed to organize a workshop on diagrammatic languages and ways to use particular cases; here's how I advertised it (from [OL18]):

> When we explain a theorem to children — in the strict sense of the term — we focus on concrete examples, and we avoid generalizations, abstract structures and infinite objects.
>
> When we present something to "children", in a wider sense of the term that means "people without mathematical maturity", or even "people without expertise in a certain area", we usually do something similar: we start from a few motivating examples, and then we generalize.
>
> One of the aims of this workshop is to discuss techniques for *particularization* and *generalization*. Particularization is easy; substituing variables in a general statement is often enough to do the job. Generalization is much harder, and one way to visualize how it works is to regard particularization as a projection: a coil projects a circle-like shadow on the ground, and we can ask for ways to "lift" pieces of that circle to the coil continously. *Projections* lose dimensions and may collapse things that were originally different; *liftings* try to reconstruct the missing information in a sensible way. There may be several different liftings for a certain part of the circle, or none. Finding good generalizations is somehow like finding good liftings.
>
> The second of our aims is to discuss *diagrams*. For example,

> in Category Theory statements, definitions and proofs can be often expressed as diagrams, and if we start with a general diagram and particularize it we get a second diagram with the same shape as the first one, and that second diagram can be used as a version "for children" of the general statement and proof. Diagrams were for a long time considered second-class entities in CT literature ([Krö07] discusses some of the reasons), and were omitted; readers who think very visually would feel that part of the work involved in understanding CT papers and books would be to reconstruct the "missing" diagrams from algebraic statements. Particular cases, even when they were the motivation for the general definition, are also treated as somewhat second-class — and this inspires a possible meaning for what can call "Category Theory for Children": to start from the diagrams for particular cases, and then "lift" them to the general case. Note that this can be done outside Category Theory too; [Jam01] is a good example.
>
> Our third aim is to discuss *models*. A standard example is that every topological space is a Heyting Algebra, and so a model for Intuitionistic Predicate Logic, and this lets us explain visually some features of IPL. Something similar can be done for some modal and paraconsistent logics; we believe that the figures for that should be considered more important, and be more well-known.

This is from the second announcement:

> If we say that categorical definitions are "for adults" - because they may be very abstract - and that particular cases, diagrams, and analogies are "for children", then our intent with this workshop becomes easy to state. "Children" are willing to use "tools for children" to do mathematics, even if they will have to translate everything to a language "for adults" to make their results dependable and publishable, and even if the bridge between their tools "for children" and "for adults" is somewhat defective, i.e., if the translation only works on simple cases...
>
> We are interested in that *bridge* between maths "for adults" and "for children" in several areas. Maths "for children" are hard to publish, even informally as notes (see this thread

http://angg.twu.net/categories-2017may02.html

in the Categories mailing list), so often techniques are rediscovered over and over, but kept restricted to the "oral culture" of the area.

Our main intents with this workshop are:

- to discuss (over coffe breaks!) the techniques of the "bridge" that we currently use in seemingly ad-hoc ways,
- to systematize and "mechanize" these techniques to make them quicker to apply,
- to find ways to publish those techniques — in journals or elsewhere,
- to connect people in several areas working in related ideas, and to create repositories of online resources.

In the UniLog 2018 I was able to chat with several categorists, and they told me about the oral culture of CT and showed me that it was not as I was guessing, and I also spent two evenings with Peter Arndt working on factorizations of geometric morphisms "for children" — and this made me feel that I could present applications of this diagrammatic language in conferences that were more top-level-ish in some sense.

The following quote is from the abstract of my submission ([MDE]) to the ACT2019:

> Imagine two category theorists, Aleks and Bob, who both think very visually and who have exactly the same background. One day Aleks discovers a theorem, T_1, and sends an e-mail, E_1, to Bob, stating and proving T_1 in a purely algebraic way; then Bob is able to reconstruct by himself Aleks's diagrams for T_1 exactly as Aleks has thought them. We say that Bob has reconstructed the *missing diagrams* in Aleks's e-mail.
>
> Now suppose that Carol has published a paper, P_2, with a theorem T_2. Aleks and Bob both read her paper independently, and both pretend that she thinks diagrammatically in the same way as them. They both "reconstruct the missing diagrams" in P_2 in the same way, even though Carol has never used those diagrams herself.

and this from my submission ([Och20]) to Diagrams 2020:

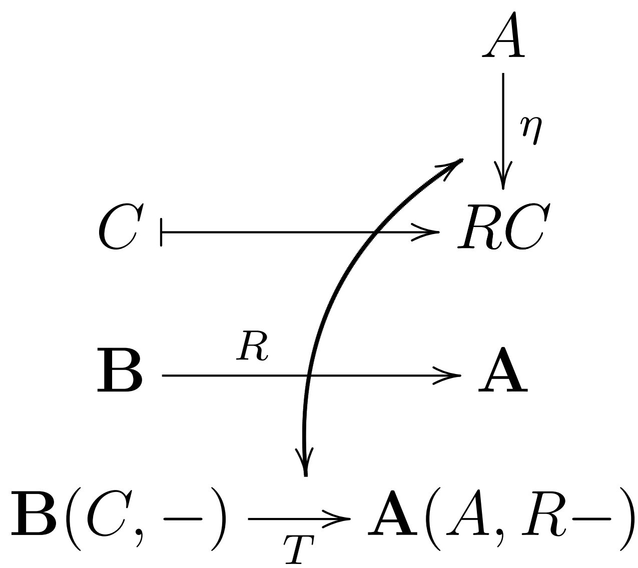
> Category Theory gives the impression of being an area where most concepts and arguments are stated and formalized via diagrams, but this is not exactly true... in most texts almost everything is done algebraically, and the reader is expected to be able to reconstruct the "missing diagrams" by himself.
>
> I used to believe, as an outsider, that some people who grew up immersed the oral culture of the area would know several techniques for "drawing the missing diagrams". My main intent when I organized the workshop "Logic for Children" at the UniLog 2018 [OL18] was to collect some of these folklore techniques, compare them with the ones that I had developed myself to study CT, and formalize them all — but what I found instead was that everybody that I could get in touch with used their own ad-hoc techniques, and that what I was trying to do was either totally new to them, or at least new in its level of detail.

The story continues in the last three sections — that also explains why I decided to write these notes using the first person in most places.

2 The conventions

The conventions that I will present now are the ones that we need for this diagram (called the "Basic Example" from here on), that is essentially the

Proposition 1 in the proof of the Yoneda Lemma in [CWM, Section III.2]:



(CD) Our diagrams are made of components that are nodes and arrows. The nodes can contain arbitrary expressions. The arrows work as connectives, and each arrow can be interpreted as the top-level connective in the smallest subexpression that contains it. For example, the curved arrow in the diagram above can be interpreted as:

$$(A \stackrel{\eta}{\to} RC) \leftrightarrow (\mathbf{B}(C,-) \stackrel{T}{\to} \mathbf{A}(A,R-)).$$

(C→) Arrows that look like '→' ("`\to`") represent hom-sets, or, in **Set**, spaces of functions. When a '→' arrow is named the name stands for an element of that hom-set. For example, in $A \stackrel{\eta}{\to} RC$ we have $\eta : A \to RC$.

(C↦) Arrows that look like '↦' ("`\mapsto`") represent internal views of functions or functors. This has some subtleties; see Section 5.

(C↔) Arrows that look like '↔' ("`\leftrightarrow`") represent bijections or isomorphisms.

(CAI) "Above" usually means "inside", or "internal view". In the diagram above the morphism $\eta : A \to RC$ is in $\mathbf{A}$ and C is an object of $\mathbf{B}$.

Also, the arrow $C \mapsto RC$ is above $\mathbf{B} \overset{R}{\to} \mathbf{A}$, and this means that it is an internal view of the functor R. Note that *usually* is not *always* — and $\mathbf{B} \overset{R}{\to} \mathbf{A}$ is not an internal view of $\mathbf{B}(C,-) \overset{T}{\to} \mathbf{A}(A,R-)$.

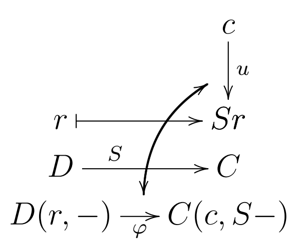
(CO) When the definition of a component of our diagram is "obvious" in the sense of "there is a unique natural construction for an object with that name", we will usually omit its definition and *pretend* that it is obvious; same for its uniqueness. See Section 3.

(CC) Everything commutes by default, and non-commutative cells have to be indicated explicitly. See Section 4.

(CTL) The default "meaning" for a diagram is the definition of its top-level component. There is a natural partial order on the components of a diagram, in which $\alpha \prec \beta$ iff α is "more basic" than β, or, in other words, if α needs to be defined before β. In the diagram above the top-level component is the curved bijection.

(CAdj) *I use shapes based on my way of drawing adjunctions whenever possible.* I like adjunctions so much that when I want to explain Category Theory to someone who knows just a little bit of Maths I always start by the adjunction $(\times B) \dashv (B\to)$ of Section 5.4; I always draw it in a canonical way, with the left adjoint going left, the right adjoint going right, and the morphisms going down. In Proposition 1 of [CWM, Section III.2] the map η is a universal arrow, and someone who learns adjunctions first sees the unit maps $\eta : A \to (B\to(A\times B))$ as the first examples of universal arrows — so that's why the upper part of the diagram above is drawn in this position.

(COT) We use a notation as close to the original text as possible, especially when we are trying to draw the missing diagrams for some existing text. If we were drawing the missing diagrams for the Proposition 1 of [CWM, Section III.2] our diagram would be this:



but I hate Mac Lane's choice of letters, so I decided to use another notation here.

(CSk) Suppose that we have a piece of text — say, a paragraph P — and we want to reconstruct the "missing diagram" D for P. Ideally this D should be a "skeleton" for P, in the sense that it should be possible to reconstruct the ideas in P from the diagram D using very few extra hints; see [IDARCT, sec.12].

(CFSh) The image by a functor of a diagram D is drawn with the same shape as D.

(CISh) The internal view of a diagram D is drawn with the same shape as D, modulo duplications — see section 5.

(CPSh) A particular case of a diagram D is drawn with the same shape as D.

(CNSh) A translation of a diagram D to another notation is drawn with the same shape as D.

Note that I have presented these conventions in a human-friendly way, that is somewhat informal and admits exceptions and extensions. Some simple examples of extensions will be discussed in Section 7.

See [Penrose] for a system that produces diagrams from conventions and specifications and then lets the user adjust these generated diagrams to make them clearer and more aesthetically pleasing — but as far as I know Penrose can only *produce* diagrams, not *read* them.

3 Finding "the" object with a given name

One of the books that I tried to read when I was starting to learn Category Theory was Mac Lane's [CWM]. It is written for readers who know a lot of mathematics and who can follow some steps that it treats as obvious. I was not (yet) a reader like that, but I wanted to become one.

There is one specific thing that [CWM] does pretending that it is obvious that I found especially fascinating. It "defines" functors by describing their actions on objects, and it leaves to the reader the task of discovering their actions on morphisms. Let's see how to find these actions on morphisms.

A functor $F : \mathbf{A} \to \mathbf{B}$ has four components:

$$F = (F_0, F_1, \mathsf{respids}_F, \mathsf{respcomp}_F).$$

They are its action on objects, its action on morphisms, the assurance that it takes identity maps to identity maps, and the assurance that it respects compositions. When Mac Lane says this,

> Fix a set B. Let $(\times B)$ denote *the* functor that takes each set A to $A \times B$.

he is saying that $(\times B)_0 A = A \times B$, or, more precisely, this:

$$(\times B)_0 := \lambda A.\, A \times B$$

The "*the*" in the expression "Let $(\times B)$ denote *the* functor..." implies that the precise meaning of $(\times B)_1$ is easy to find, and that it is easy to prove $\mathsf{respids}_{(\times B)}$ and $\mathsf{respcomp}_{(\times B)}$.

If $f : A' \to A$ then $(\times B)_1 f : (\times B)_0 A' \to (\times B)_0 A$. We know the *name* of the image morphism, $(\times B)_1 f$, and its *type*,

$$(\times B)_1 f : A' \times B \to A \times B,$$

and it is implicit that there is an "obvious" natural construction for this $(\times B)_1 f$ from f. A natural construction is — TA-DAAAA!!! — a λ-term, so we are looking for a term of type $A' \times B \to A \times B$ that can be constructed from $f : A' \to A$.

In a big diagram:

$$\dfrac{f : A' \to A}{(\times B)_1 f : A'{\times}B \to A{\times}B} \qquad\Rightarrow\qquad \dfrac{\dfrac{\dfrac{\dfrac{[p : A'{\times}B]^1}{\pi p : A'} \quad f : A' \to A}{f(\pi p) : A} \qquad \dfrac{[p : A'{\times}B]^1}{\pi' p : B}}{(f(\pi p), \pi' p) : A{\times}B}}{(\lambda p{:}A'{\times}B.(f(\pi p), \pi' p) : A'{\times}B \to A{\times}B}\;1$$

A double bar in a derivation means "there are several omitted steps here", and sometimes a double bar suggests that these omitted steps are obvious. The derivation on the left says that there is an "obvious" way to build a $(\times B)_1 f : A'{\times}B \to A{\times}B$ from a "hypothesis" $f : A' \to A$. If we expand its double bar we get the tree at the right, that shows that the "precise meaning" for $(\times B)_1 f$ is $(\lambda p{:}A'\times B.(f(\pi p), \pi' p)$. More formally (and erasing a typing),

$$(\times B)_1 := \lambda f.(\lambda p.(f(\pi p), \pi' p)).$$

The expansion of the double bar above becomes something more familiar if we translate the trees to Logic using Curry-Howard:

$$\dfrac{P \to Q}{\overline{P \land R \to Q \land R}} \qquad \Rightarrow \qquad \dfrac{\dfrac{\dfrac{\dfrac{[P \land R]^1}{P} \quad P \to Q}{Q} \qquad \dfrac{[P \land R]^1}{R}}{Q \land R}}{P \land R \to Q \land R}\,1$$

We obtain the tree at the right by *proof search*.

Let's give a name for the operation above that obtained a term of type $A'\times B \to A\times B$: we will call that operation *term search*, or, as it is somewhat related to type inference, *term inference*.

Term search may yield several different construction and trees, and so several non-equivalent terms of the desired type. When Mac Lane says "*the* functor $(\times B)$" he is indicating that:

- a term for $(\times B)_1$ is easy to find (note that we use the expression "a *precise meaning* for $(\times B)_1$"),
- all other natural constructions for something that "deserves the name" $(\times B)_1$ yield terms equivalent to that first, most obvious one,
- proving $\mathsf{respids}_{(\times B)}$ and $\mathsf{respcomp}_{(\times B)}$ is trivial.

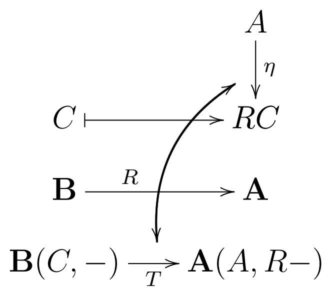
In many situations we will start by just the name of a functor, as the "$(\times B)$" in the example above, and from that name it will be easy to find *the* "precise meaning" for $(\times B)_0$, and from that the "precise meaning" for $(\times B)_1$, and after that proofs that $\mathsf{respids}_{(\times B)}$ and $\mathsf{respcomp}_{(\times B)}$. We will use the expression "...deserving the name..." in this process — terms for $(\times B)_0$, $(\times B)_1$, $\mathsf{respids}_{(\times B)}$, and $\mathsf{respcomp}_{(\times B)}$ "deserve their names" if they obey the expected constraints.

For a more thorough discussion see [IDARCT].

Note: I am not aware of any papers or books that discuss how to (re)construct a functor from its action on objects, or from its name. If you have any references, please let me know!

These ideas of "finding a precise meaning" and "finding (something) deserving that name" can also be applied to morphisms, natural transformations, isomorphisms, and so on.

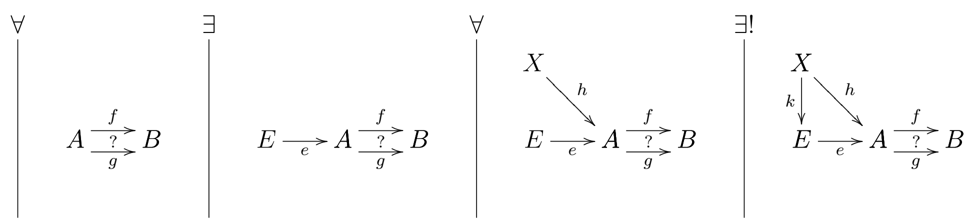
In Section 6.3 we will see how to find natural constructions for the two directions of the bijection in the Basic Example — or how the expand the double bars in the two derivations here:



$$\dfrac{\gamma : A \to RC}{T : \mathbf{B}(C,-) \to \mathbf{A}(A,R-)} \qquad \dfrac{T : \mathbf{B}(C,-) \to \mathbf{A}(A,R-)}{\gamma : A \to RC}$$

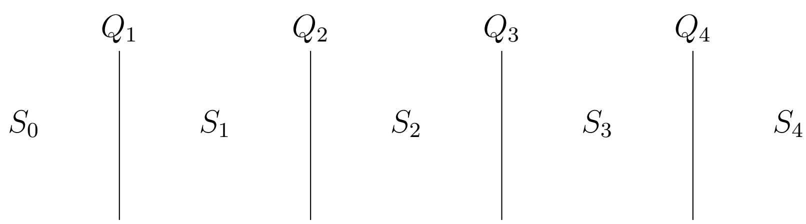
4 Freyd's diagrammatic language

In [Freyd76] Peter Freyd presents a very nice diagrammatic language that can be used to express *some* definitions from Category Theory. For example, this is the statement that a category has all equalizers:



All cells in these diagrams commute by default, and non-commuting cells have to be indicated with a '?'. Each vertical bar with a '∀' above it means "for all extensions of the previous diagram to this one such that everything commutes"; a vertical bar with a '∃!' above it means "there exists a unique extension of the previous diagram to this one such that everything commutes", and so on. See the scan in [Freyd76] for the basic details of how to formalize these diagrams, and the book [FS90, p.28 onwards], for tons of extra details, examples, and applications.

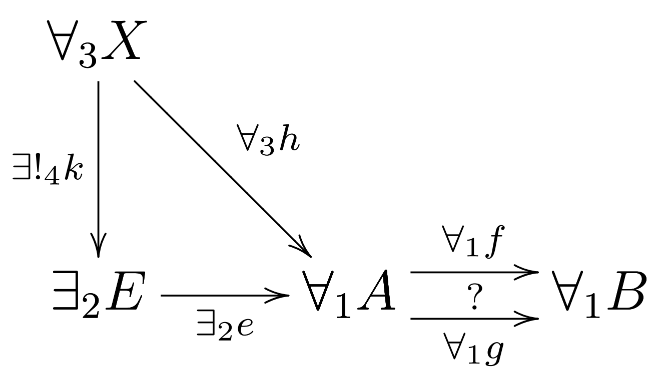
Let's call the subdiagrams of a diagram like the one above its "stages". Its stage 0 is empty, its stage 1 has two objects and two arrows, its last stage has four objects and five arrows, and the quantifiers separating the stages are $Q_1 = \forall$, $Q_2 = \exists$, $Q_3 = \forall$, $Q_4 = \exists!$. They are structured like this:



I was not very good at drawing all stages separately — it was boring, it took me too long, and I often got distracted and committed errors — so I started to play with extensions of that diagrammatic language.

4.1 Adding quantifiers

Here is a simple way to draw all stages at once. We start from a diagram for the "last stage with quantifiers", that we will call LSQ:



We can recover all the stages and quantifiers from it. The numbered quantifiers in it are $\forall_1$, $\exists_2$, $\forall_3$, and $\exists!_4$. The highest number in them 4, so we set $n = 4$ (n is the index of the last stage), and we set "stage 4 with quantifiers", SQ_4, to LSQ. To obtain the SQ_3 from SQ_4 we delete all nodes an arrows in SQ_4 that are annotated with a '$\exists!_4$'; to obtain SQ_2 from SQ_3

we delete all nodes an arrows in SQ_3 that are annotated with a '$\forall_3$', and so on until we get a diagram SQ_0, that in this example is empty. To obtain each S_k — a stage in the original diagrammatic language from Freyd, that doesn't have quantifiers — from the corresponding SQ_k we treat all the quantifiers in SQ_k as mere annotations, and we erase them; for example, '$\exists_2 e$' becomes 'e', and $\forall_1 A$ becomes A. To obtain the quantifiers Q_1, Q_2, Q_3, Q_4 that are put in the vartical bars that separate the stages, we just assign $\forall_1$, $\exists_2$, $\forall_3$, and $\exists!_4$ to them, without the numbers in the subscripts.

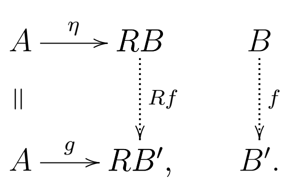
Bonus convention: when the quantifiers in a diagram are just '$\forall$'s and '$\exists!$'s without subscripts the '$\forall$'s are to be interpreted as '$\forall_1$' and the '$\exists!$'s as '$\exists!_2$'s.

4.2 Adding functors

Freyd's language can't represent functors[1], and I wanted to use it to draw the missing diagrams for definitions involving functors, so I had to extend it again.

Let me use an example to discuss this. This is the definition of universal arrow in [CWM, p.55], including the original diagram, modulo change of letters:

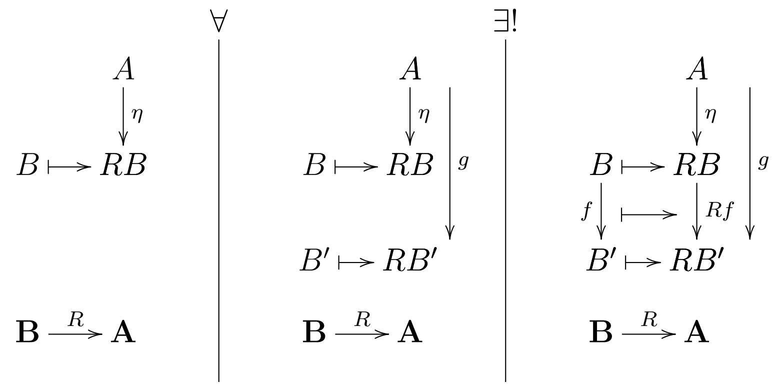
> **Definition.** If $R : \mathbf{B} \to \mathbf{A}$ is a functor and A an object of $\mathbf{A}$, a universal arrow from A to R is a pair (B, η) consisting of an object B of $\mathbf{B}$ and and arrow $\eta : A \to RB$ of $\mathbf{A}$ such that to every pair (B', g) with B' an object of $\mathbf{B}$ and $g : A \to RB'$ an arrow of $\mathbf{A}$, there is a unique arrow $f : B \to B'$ of $\mathbf{B}$ with $Rf \circ \eta = g$. In other words, every arrow h to R factors uniquely through the universal arrow η, as in the commutative diagram:



The definition itself goes only up to the "with $Rf \circ \eta = g$.", so let me ignore the part starting from "In other words", and draw a better "missing

[1] As far as I know — I don't know [FS90] very well.

diagram” for the definition:

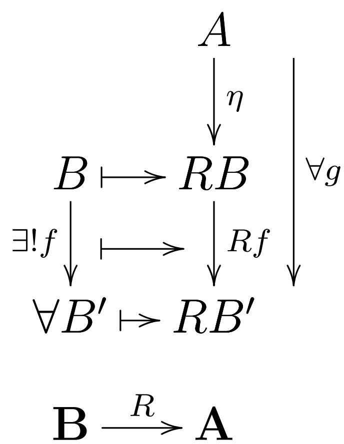


This diagram is quite close to being a skeleton for the definition of universal arrow. It can be interpreted as a proposition, and the only extra hint that we need is that “universalness” for the arrow η corresponds to the truth of that proposition. Here’s how to extract the proposition from it:

In a context where:	$\mathbf{A}$ is a category,
	$\mathbf{B}$ is a category,
	$R : \mathbf{B} \to \mathbf{A}$,
	$A \in \mathbf{A}$,
	$B \in \mathbf{B}$,
	$\eta : A \to RB$,
for all	$B' \in \mathbf{B}$ and
	$g : A \to RB'$,
there exists a unique	$f : B \to B'$ such that
	$Rf \circ \eta = g$.

To convert that to a definition of universalness we just have to replace the “for all” by “(B, η) is a universal arrow for A to R iff for all”.

The convention for quantifiers from sec.4.1 lets us rewrite the diagram in three stages above as:



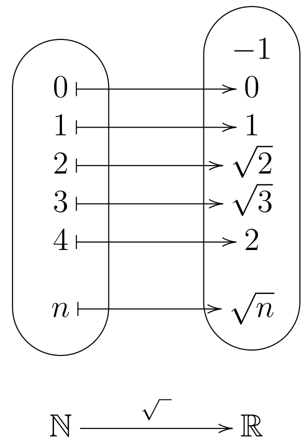
Also, I noticed that I could omit most typings when they could be inferred from the diagram. I could "formalize" the diagram above as: "in a context where $(\mathbf{A}, \mathbf{B}, R, A, B, \eta)$ are as in the diagram above, we say that (B, η) is a universal arrow from A to R when $\forall(B', g).\exists! f.(Rf \circ \eta = g)$". This may be too loaded to be used in public, but it's very practical for private notes — and I can even omit the "$Rf \circ \eta = g$", as everything commutes by default.

Note that when we erase a node or arrow we also erase everything that depends on it. In the example above SQ_2 has an arrow labeled $\exists!_2 f$; to obtain SQ_1 from SQ_2 we have to erase that arrow, the arrow Rf, and the arrow $f \mapsto Rf$ — and to obtain SQ_0 from SQ_1 we have to erase the arrow g, the node B', the node RB', and the arrow $B' \mapsto RB'$.

5 Internal views

My favorite way of introducing internal views is with the diagram below:

$$\begin{array}{rccc} \sqrt{\ } : & \mathbb{N} & \to & \mathbb{R} \\ & n & \mapsto & \sqrt{n} \end{array}$$



The parts with the two blobs and '$\longmapsto$'s between them is based on how I was taught sets and functions when I was a kid; it is an internal view of the $\mathbb{N} \stackrel{\sqrt{\ }}{\to} \mathbb{R}$ below it. Not all elements of ℕ are shown in the blob-view of ℕ, but the ones that are shown are named; compare this with [LR03, p.2 onwards], in which the elements are usually dots.

The arrow $n \longmapsto \sqrt{n}$ between the blobs shows a *generic element* of ℕ and its image, and the other '$\longmapsto$'s are *substitution instances of it*, like this:

$$(n \longmapsto \sqrt{n})[n := 2] = (2 \longmapsto \sqrt{2})$$

In some cases, like $4 \longmapsto 2$, we write 2 instead of $\sqrt{4}$ because $\sqrt{4}$ "reduces to" 2, as explained in the next section.

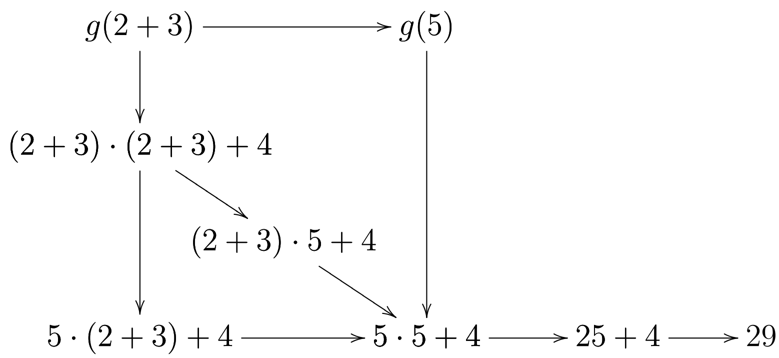
5.1 Reductions

The convention (C↦) says that an arrow $\alpha \mapsto \beta$ above an arrow $A \stackrel{f}{\to} B$ should be interpreted as meaning $f(\alpha) \rightsquigarrow \beta$, where '$\rightsquigarrow$' means "reduces to"; the standard example is $\sqrt{4} \rightsquigarrow 2$. In a diagram:

$$\begin{array}{ll} 4 \longmapsto 2 & \sqrt{4} \rightsquigarrow 2 \\ n \longmapsto \sqrt{n} & \\ \\ \mathbb{N} \xrightarrow{\sqrt{\ }} \mathbb{R} & \end{array} \qquad \begin{array}{ll} \alpha \longmapsto \beta & f(\alpha) \rightsquigarrow \beta \\ \\ A \xrightarrow{f} B & \end{array}$$

The idea of reduction comes from λ-calculus. We write $\alpha \overset{1}{\rightsquigarrow} \beta$ to say that the term α reduces to β in one step, and $\alpha \overset{*}{\rightsquigarrow} \gamma$ to say that there is a finite sequence of one-step reductions that reduce α to γ. Here we are interested in reduction in a system with constants, in which for example $(\sqrt{\ })(4) \overset{1}{\rightsquigarrow} 2$.

Here is a directed graph that shows all the one-step reductions starting from $g(2+3)$, considering $g(a) = a \cdot a + 4$:

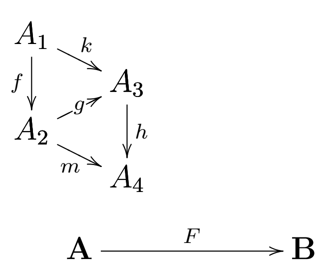


Note that all reductions sequences starting from $g(2+3)$ terminate at the same term, 29 — "the term $g(2+3)$ is strongly normalizing" —, and reduction sequences from $g(2+3)$ may "diverge" but they "converge" later — this is the "Church-Rosser Property", a.k.a. "confluence".

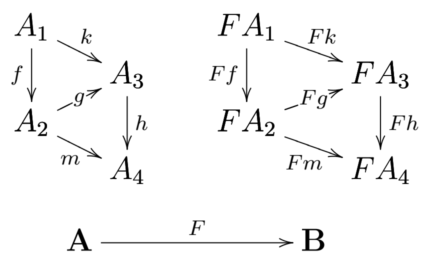
A good place to learn about reduction in systems with constants is [SICP].

5.2 Functors

By the convention (CFSh) the image of the diagram above **A** in the diagram below — remember that *above* usually means *inside* —

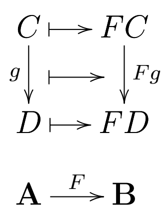


is a diagram with the same shape over $\mathbf{B}$. We draw it like this:



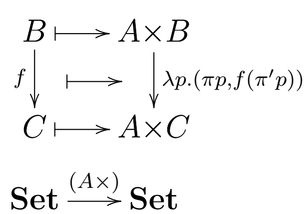
In this case we don't draw the arrows like $A_1 \mapsto FA_1$ because there would be too many of them — we leave them implicit.

We say that the diagram above is *an* internal view of the functor F. To draw *the* internal view of the functor $F : \mathbf{A} \to \mathbf{B}$ we start with a diagram in $\mathbf{A}$ that is made of two generic objects and a generic morphism between them. We get this:



Compare this with the diagram with blob-sets in Section 5, in which the '$n \mapsto \sqrt{n}$' says where a generic element is taken.

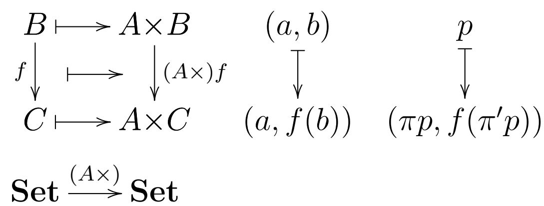
Any arrow of the form $\alpha \mapsto \beta$ above a functor arrow $\mathbf{A} \xrightarrow{F} \mathbf{B}$ is interpreted as saying that F takes α to β, or, in the terminology of the section 5.1, that $F\alpha$ reduces to β. So this diagram



defines $(A\times)$ as:

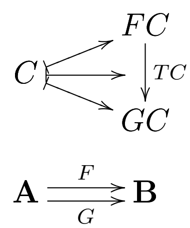
$$\begin{array}{rcl} (A\times)_0 & := & \lambda B.\, A \times B, \\ (A\times)_1 & := & \lambda f.\lambda p.(\pi p, f(\pi' p)). \end{array}$$

In this case we can also use internal views of $(A\times)$ to define $(A\times)_1$:

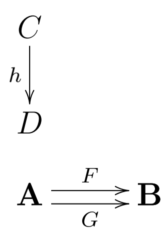


5.3 Natural transformations

Suppose that we have two functors $F, G : \mathbf{A} \to \mathbf{B}$ and a natural transformation $T : F \to G$. A first way to draw an internal view of T is this:

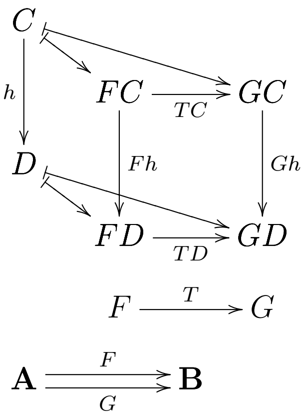


If we start with a morphism $h : C \to D$ in $\mathbf{A}$, like this,

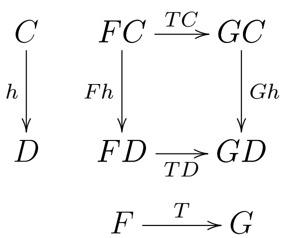


the convention (CFSh) would yield an image of h by F and another by G, and we can draw the arrows TC and TD to obtain a commuting square in

B:

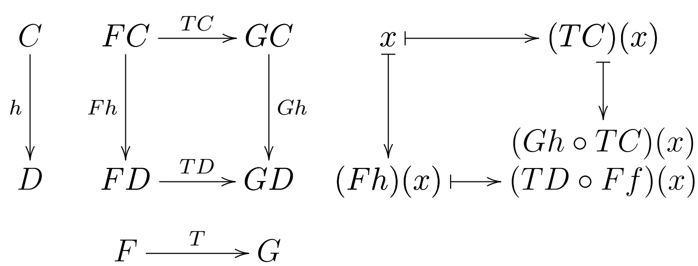


This way of drawing internal views of natural transformations yields diagrams that are too heavy, so we will usually draw them as just this:



Note that the input morphism is at the left, and above $F \xrightarrow{T} G$ we draw its images by F, G, and T.

When the codomain of F and G is **Set** we will sometimes also draw at the right an internal view of the commuting square, like this:

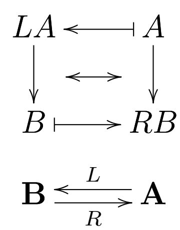


Then the commutativity of the middle square is equivalent to $\forall x \in FC.(Gh \circ TC)(x) = (TD \circ Ff)(x)$. Note that in this case the square at the right is an internal view of an internal view.

In Section 3 we saw that a functor has four components. A natural transformation has two: $T = (T_0, \mathsf{sqcond}_T)$, where T_0 is the operation $C \mapsto TC$ and sqcond_T is the guarantee that all the induced squares commute. Sometimes we will use the upper line of the internal view of the internal view to define T_0 — see Section 6.2 for an example of this.

5.4 Adjunctions

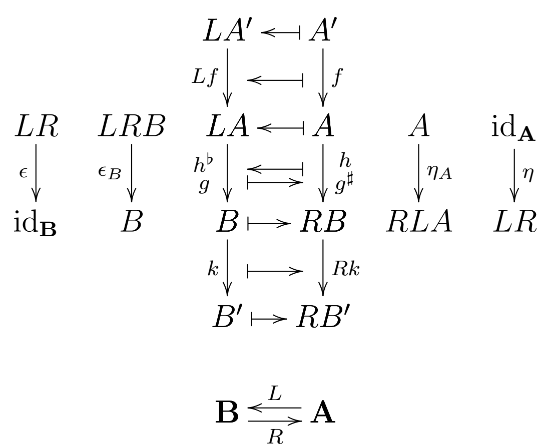
We will draw adjunctions like this,



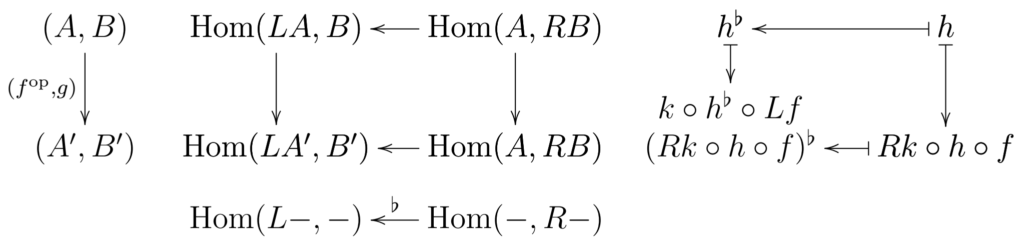
with the left adjoint going left and the right adjoint going right. My favorite names for the left and right adjoints are L and R. The standard notation for that adjunction is $L \dashv R$.

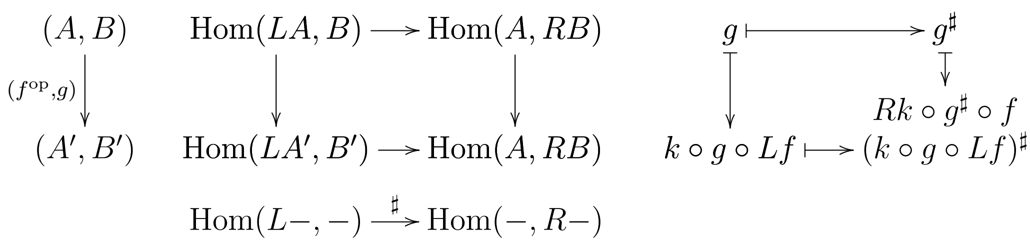
The top-level component of the diagram above is the bijection arrow in the middle of the square — it says that $\mathrm{Hom}(LA, B) \leftrightarrow \mathrm{Hom}(A, RB)$. It is implicit that we have bijections like that for all A and B; it is also implicit that that bijection is natural in some sense.

We will sometimes expand adjunction diagrams by adding unit and counit maps, the unit and the unit as natural transformations, the actions of L and R on morphisms, and other things. For example:



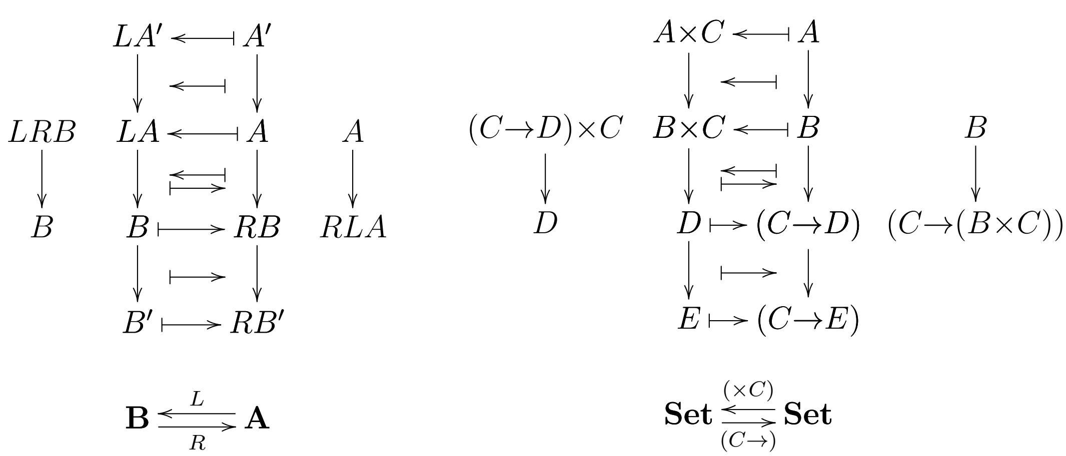
We can obtain the naturality conditions by regarding $\flat$ and $\sharp$ as natural transformations and drawing the internal views of their internal views:





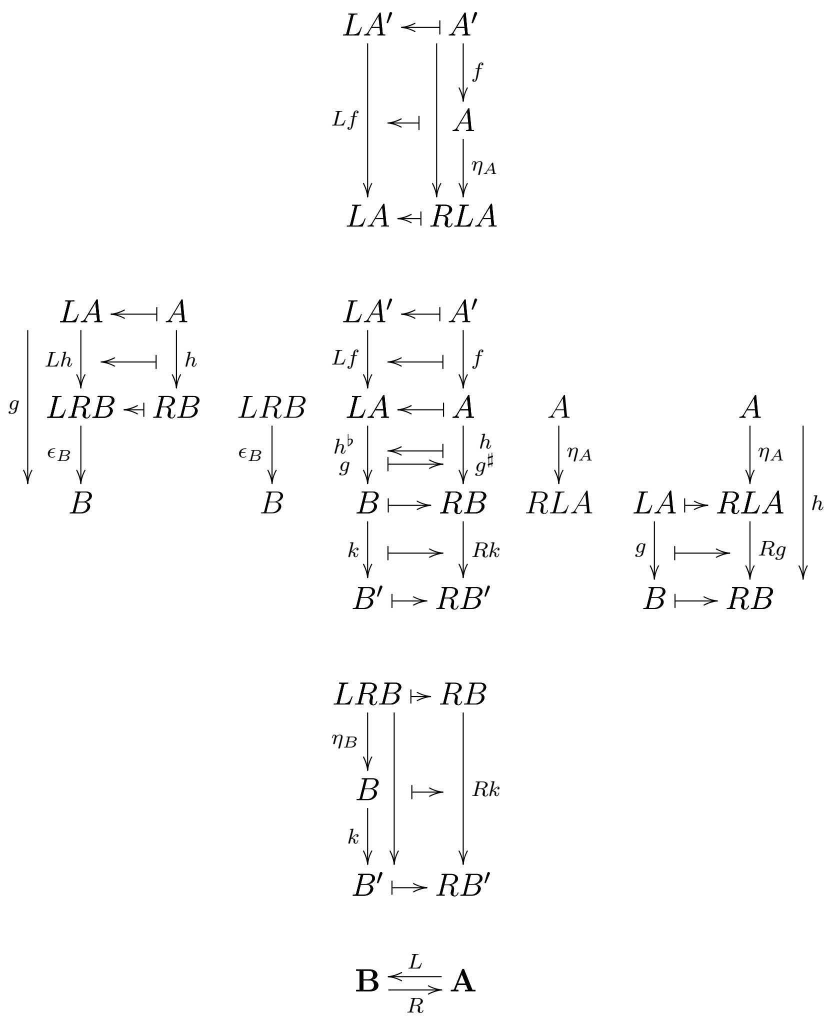
5.5 A way to teach adjunctions

I mentioned in the first sections that I have tested some parts of this language in a seminar course — described here: [Och19] — and that in it I teach Categories starting by adjunctions. Here's how: we start by the basics of λ-calculus and some sections of [PH1], and then I ask the students to define each one of the operations in the right half of the diagram below as λ-terms:



Then we see the definition of functors, natural transformations and adjunctions, and we check that the right half is a particular case of the diagram for a generic adjunction in the left half. After that, and after also checking that in the Planar Heyting Algebras of [PH1] we have an adjunction $(\wedge Q) \dashv (Q \to)$, I help the students to decypher some excerpts of standard texts on CT — in the last time that I gave the course we used [Awo06], but I am planning to use [CWM] the next time.

From the components of the generic adjunction in the diagram above it is possible to build this big diagram:



Let's use these names for its subdiagrams: $\begin{smallmatrix} & & A & & \\ B & C & D & E & F \\ & & G & & \\ & & I & & \end{smallmatrix}$.

A *fully-specified adjunction* between categories $\mathbf{B}$ and $\mathbf{A}$ has lots of components: $(L, R, \epsilon, \eta, \flat, \sharp, \mathsf{univ}(\epsilon), \mathsf{univ}(\eta))$, and maybe even others, but usually we define only some of these components; there is a Big Theorem About Adjunctions (below!) that says how to reconstruct the fully-specified adjunction from some of its components.

Some parts of the diagram above can be interpreted as definitions, like these:

$$Lf := (\eta_A \circ f)^\flat$$

$$g := \epsilon_B \circ Lh \qquad \epsilon_B := (\mathrm{id}_{RB})^\flat \qquad \eta_A := (\mathrm{id}_{LA})^\sharp \qquad h := Rg \circ \eta_A$$

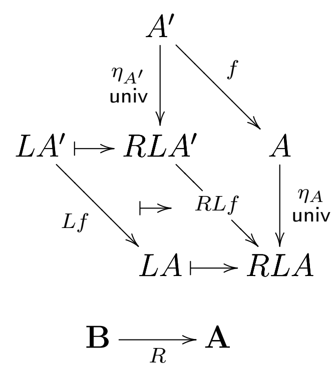
$$Rk := (k \circ \eta_B)^\sharp$$

The subdiagrams B and F can also be interpreted in the opposite direction, as:

$$\begin{array}{rcl} g^\sharp & := & (\forall A.\forall g.\exists!h)Ag \\ & = & (\mathsf{univ}_{\epsilon_B})Ag \end{array} \qquad \begin{array}{rcl} h^\flat & := & (\forall B.\forall h.\exists!g)Bh \\ & = & (\mathsf{univ}_{\eta_A})Bh \end{array}$$

The notations $(\forall A.\forall g.\exists!h)Ag$ and $(\mathsf{univ}_{\epsilon_B})Ag$ are clearly abuses of language — but it's not hard to translate them to something formal, and they inspire great discussions in the classroom... also, they can help us to understand and formalize constructions like this one,

$$Lf := (\mathsf{univ}_{\eta_A})(LA)(\eta_A \circ f)$$



that are needed in cases like the part (ii) of the Big Theorem.

The Big Theorem About Adjunctions is this — it's the Theorem 2 in [CWM, page 83], but with letters changed to match the ones we are using in our diagrams:

Big Theorem About Adjunctions. Each adjunction $\langle L, R, \sharp \rangle : \mathbf{A} \rightharpoonup \mathbf{B}$ is completely determined by the items in any one of the following lists:

(i) Functors L, R, and a natural transformation $\eta : \mathrm{id}_{\mathbf{A}} \to RL$ such that each $\eta_A : A \to RLA$ is universal to R from A. Then $\sharp$ is defined by (6).

(ii) The functor $R : \mathbf{B} \to \mathbf{A}$ and for each $A \in \mathbf{A}$ an object $L_0A \in \mathbf{B}$ and a universal arrow $\eta_A : A \to RL_0A$ from A to R. Then the functor L has object function L_0 and is defined on arrows $f : A' \to A$ by $RLf \circ \eta_{A'} = \eta_A \circ f$.

(iii) Functors L, R, and a natural transformation $\epsilon : LR \to \mathrm{id}_{\mathbf{B}}$ such that each $\epsilon_B : LRB \to B$ is universal from L to B. Here $\flat$ is defined by (7).

(iv) The functor $L : \mathbf{A} \to \mathbf{B}$ and for each $B \in \mathbf{B}$ an object $R_0B \in \mathbf{A}$ and an arrow $\epsilon_B : LR_0B \to B$ universal from L to B.

(v) Functors L, R and natural transformations $\eta : \mathrm{id}_{\mathbf{A}} \to RL$ and $\epsilon : LR \to \mathrm{id}_{\mathbf{B}}$ such that both composites (8) are the identity transformations. Here $\sharp$ is defined by (6) and $\flat$ by (7).

My plan for the next incarnation of the course is to ask the students to 1) visualize in the big diagram all the objects and constructions in the Big Theorem, 2) take the original Theorem 2 in [CWM] and draw the missing diagrams for it, 3) decypher some other parts of the section about adjunctions in [CWM].

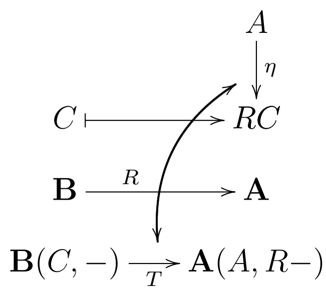
6 The Basic Example as a skeleton

In the sections 2 and 3 I claimed that the diagram of the Basic Example is a “skeleton” of a certain theorem, in the sense that both the statement and the proof of that theorem can be reconstructed from just the diagram and very few extra hints. Let’s see the details of this.

6.1 Reconstructing its functors

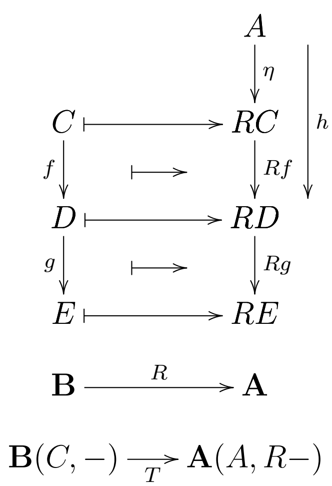
Let’s call this diagram — the diagram of the Basic Example — Y0:

Y0 :=



We don’t know yet the precise meaning of the functors $\mathbf{B}(C, -)$ and $\mathbf{A}(A, R-)$, but if we enlarge Y0 to

Y0$^+$:=



and we draw the internal views of $\mathbf{B}(C, -)$ and $\mathbf{A}(A, R-)$ then the meanings for $\mathbf{B}(C,-)$ and $\mathbf{A}(A,R-)$ become obvious:

$$\begin{array}{ccccccc}
D & \mapsto & \mathbf{B}(C,D) & f & D & \mapsto & \mathbf{A}(A,RD) & h \\
{\scriptstyle g}\downarrow & & \downarrow{\scriptstyle \mathbf{B}(C,g)} & \downarrow & {\scriptstyle g}\downarrow & & \downarrow{\scriptstyle \mathbf{A}(A,Rg)} & \downarrow \\
E & \mapsto & \mathbf{B}(C,E) & g\circ f & E & \mapsto & \mathbf{A}(A,RE) & Rg\circ h \\
\mathbf{B} & \xrightarrow{\mathbf{B}(C,-)} & \mathbf{Set} & & \mathbf{B} & \xrightarrow{\mathbf{A}(A,R-)} & \mathbf{Set} &
\end{array}$$

So:

$$\begin{array}{rcl}
\mathbf{B}(C,-) & : & \mathbf{B}\to\mathbf{Set} \\
\mathbf{B}(C,-)_0 & := & \lambda D.\mathbf{B}(C,D) \\
\mathbf{B}(C,-)_1 & := & \lambda g.\lambda f.g\circ f \\
\mathbf{A}(A,R-) & : & \mathbf{B}\to\mathbf{Set} \\
\mathbf{A}(A,R-)_0 & := & \lambda D.\mathbf{A}(A,RD) \\
\mathbf{A}(A,R-)_1 & := & \lambda g.\lambda h.Rg\circ h
\end{array}$$

6.2 Reconstructing its natural transformation

We also don't know — yet — what is the natural transformation

$$\mathbf{B}(C,-)\xrightarrow{T}\mathbf{A}(A,R-).$$

Its internal view is this:

$$\begin{array}{cccccc}
D & \mathbf{B}(C,D) & \xrightarrow{TD} & \mathbf{A}(A,RD) & f & h \\
{\scriptstyle g}\downarrow & {\scriptstyle \mathbf{B}(C,g)}\downarrow & & \downarrow{\scriptstyle \mathbf{A}(A,Rg)} & \downarrow & \downarrow \\
E & \mathbf{B}(C,E) & \xrightarrow{TE} & \mathbf{A}(A,RE) & g\circ f & Rg\circ h \\
 & \mathbf{B}(C,-) & \xrightarrow{T} & \mathbf{A}(A,R-) & &
\end{array}$$

Note that we only drew the vertical arrows of the internal view of the internal view.

If we have an arrow $\eta: A\to RC$ then we have a natural construction for T_0: $TD(f) := Rf\circ\eta$, and we can redraw the internal view of the internal view as:

$$\begin{array}{cccc}
f & \mapsto & Rf\circ\eta & h \\
\downarrow & & \downarrow & \downarrow \\
 & & Rg\circ(Rf\circ\eta) & Rg\circ h \\
g\circ f & \mapsto & R(g\circ f)\circ\eta &
\end{array}$$

The square condition clearly holds, because:

$$\begin{array}{rcl} Rg \circ (Rf \circ \eta) & = & (Rg \circ Rf) \circ \eta \\ & = & R(g \circ f) \circ \eta. \end{array}$$

So

$$T_0 \quad := \quad \lambda D.\lambda f.Rf \circ \eta.$$

6.3 Reconstructing its bijection

We can give names like 'd' and 'u' for the two components of the curved bijection, like this:

$$\begin{array}{c} \mathrm{Hom}(A, RC) \\ {\scriptstyle d}\downarrow\uparrow{\scriptstyle u} \\ \mathrm{Hom}(\mathbf{B}(C,-), \mathbf{A}(A, R-)) \end{array} \qquad \begin{array}{cc} \eta & \eta \\ \downarrow & \uparrow \\ T & T \end{array} \qquad \begin{array}{cc} \eta & u(T) \\ \downarrow & \uparrow \\ d(\eta) & T \end{array} \qquad \begin{array}{cc} \eta & \eta_T \\ \downarrow & \uparrow \\ T_\eta & T \end{array}$$

but the notation at the right will be clearer.

We just saw how the direction 'd' of the bijection works:

$$(T_\eta)_0 \quad := \quad \lambda D.\lambda f.Rf \circ \eta.$$

Here's how to find a natural construction for u. Suppose that we have a natural transformation T. Then $TC(\mathrm{id}_C)$ is an element of $\mathbf{A}(A, RC)$:

$$\begin{array}{lll} C & \mathbf{B}(C,C) \xrightarrow{TC} \mathbf{A}(A, RC) & \mathrm{id}_C \longmapsto TC(\mathrm{id}_C) \\ & \mathbf{B}(C,-) \xrightarrow{T} \mathbf{A}(A, R-) & \end{array}$$

We can define:

$$\eta_T \quad := \quad TC(\mathrm{id}_C).$$

Now we need to check that d and u are mutually inverse, or, in the other notation, that the round trips $\eta \mapsto T_\eta \mapsto \eta_{(T_\eta)}$ and $T \mapsto \eta_T \mapsto T_{(\eta_T)}$ are identity maps. Here is a good way to draw the round trips:

$$\begin{array}{cc} \eta & \eta_{(T_\eta)} \\ \downarrow & \uparrow \\ T_\eta & T_\eta \end{array} \qquad \begin{array}{cc} \eta_T & \eta_T \\ \downarrow & \uparrow \\ T_{(\eta_T)} & T \end{array}$$

Checking that $\eta \mapsto T_\eta \mapsto \eta_{(T_\eta)}$ yields back the original η is easy — we just have to start with $\eta_{(T_\eta)}$ and reduce it as most as we can:

$$\begin{array}{rcl}
\eta_{(T_\eta)} & = & T_\eta C(\mathrm{id}_C) \\
 & = & (\lambda D.\lambda g.(Rg \circ \eta))C(\mathrm{id}_C) \\
 & = & (\lambda g.(Rg \circ \eta))(\mathrm{id}_C) \\
 & = & R(\mathrm{id}_C) \circ \eta \\
 & = & \mathrm{id}_{RC} \circ \eta \\
 & = & \eta
\end{array}$$

Checking that the other round trip, $T \mapsto \eta_T \mapsto T_{(\eta_T)}$, yields back the original T is not trivial. In the terminology of the convention (CSk) from Section 2, to reconstruct that proof we need an extra hint: that at some point in the proof we will have to use that the original T obeys sqcond_T, and that we will have to "evaluate" sqcond_T on these inputs:

$$\begin{array}{ccc}
C & & \mathrm{id}_C \\
{\scriptstyle f}\downarrow & & \\
D & & \\
 & \cdot \xrightarrow{T} \cdot &
\end{array}$$

This yields:

$$\begin{array}{cccccc}
C & \mathbf{B}(C,C) \xrightarrow{TC} \mathbf{A}(A,RC) & \mathrm{id}_C & \longmapsto & TC(\mathrm{id}_C) \\
{\scriptstyle f}\downarrow & {\scriptstyle \mathbf{B}(C,f)}\downarrow \qquad\qquad \downarrow{\scriptstyle \mathbf{A}(A,Rf)} & \downarrow & & \downarrow \\
 & & & & Rf \circ (TC(\mathrm{id}_C)) \\
D & \mathbf{B}(C,D) \xrightarrow{TD} \mathbf{A}(A,RD) & f \circ \mathrm{id}_C = f & \longmapsto & TD(f) \\
 & & & & \\
 & \mathbf{B}(C,-) \xrightarrow{T} \mathbf{A}(A,R-) & & &
\end{array}$$

so $Rf \circ (TC(\mathrm{id}_C)) = TD(f)$.

We want to check that for all D and f we have $T_{(\eta_T)}D(f) = TD(f)$. We have:

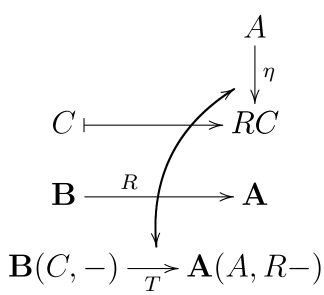
$$\begin{array}{rcl}
T_{(\eta_T)}D(f) & = & (\lambda D.\lambda f.Rf \circ \eta_T)D(f) \\
 & = & (\lambda f.Rf \circ \eta_T)(f) \\
 & = & Rf \circ \eta_T \\
 & = & Rf \circ (TC(\mathrm{id}_C)) \\
 & = & TD(f).
\end{array}$$

It works! So we have a natural construction for the bijection $T \leftrightarrow \eta$, given by:

$$\begin{array}{rcl} T_0 & := & \lambda D.\lambda f.Rf \circ \eta \\ \eta & := & TC(\mathrm{id}_C) \end{array}$$

6.4 The full reconstruction

We have just reconstructed all the typings and definitions for the diagram Y0. Here is the full reconstruction, except for the "proof terms" like respids, assoc, idL and idR for each functor, sqcond for each natural transformations, and the proofs that both round trips in the bijections are identity maps:



$\mathbf{A}$ is a category,
$\mathbf{B}$ is a category,
$R : \mathbf{B} \to \mathbf{A}$,
$A \in \mathbf{A}$,
$C \in \mathbf{B}$,
$\eta : A \to RD$,

$\mathbf{B}(C, -) : \mathbf{B} \to \mathbf{Set}$,
$\mathbf{B}(C, -)_0 := \lambda D.\mathbf{B}(C, D)$,
$\mathbf{B}(C, -)_1 := \lambda g.\lambda f.g \circ f$,

$\mathbf{A}(A, R-) : \mathbf{A} \to \mathbf{Set}$,
$\mathbf{A}(A, R-)_0 := \lambda D.\mathbf{A}(A, RD)$,
$\mathbf{A}(A, R-)_1 := \lambda g.\lambda h.Rg \circ h$,

$T : \mathbf{B}(C, -) \to \mathbf{A}(A, R-)$,

$T_0 := \lambda D.\lambda f.Rf \circ \eta$,
$\eta := TC(\mathrm{id}_C)$.

It shouldn't be hard — for someone with practice — to translate the types and definitions at the right above to the language of some proof assistant. I tried to do this in Idris ([Bra17]) using [IdrisCT] but I didn't go very far... I implemented the protocategories, protofunctors and proto-NTs of [IDARCT, section 19] to be able to skip the proof terms on my first prototypes, but I got stuck trying to implement the formalization of Y0 as a single datatype...

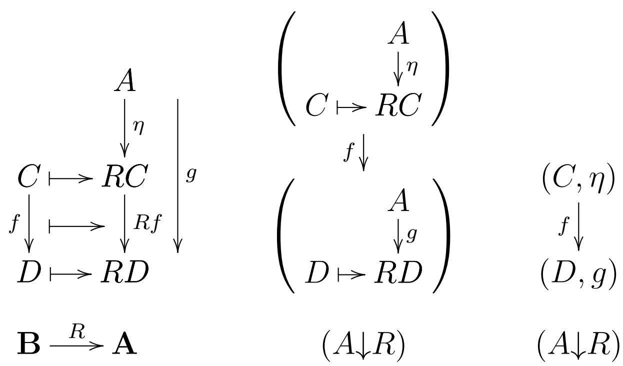
(Help would be greatly appreciated!...)

7 Extensions to the diagrammatic language

Our diagrammatic language and the list of conventions in Section 2 can be extended — "by the user" — in zillions of ways. Let's see some examples of extensions.

7.1 A way to define new categories

We saw in the sections 5.2 and 6.1 how to use diagrams to define functors, and in sections 5.3 and 6.2 how to define natural transformations. We can define new categories by diagrams, too.



My favorite way — a syntax sugar! — of visualizing the comma category $(A{\downarrow}R)$ is the middle third of the diagram above, in which the objects of $(A{\downarrow}R)$ are depicted as L-shaped diagrams. To understand the typings and the commutativity conditions we have to look at the left third — it indicates that f must obey $Rf \circ \eta = g$. The right third shows a generic morphism in $(A{\downarrow}R)$ without the syntax sugar, but we still have to look at the left third to type it. We have:

In a context in which	$\mathbf{A}$ is a category,
	$\mathbf{B}$ is a category,
	$R : \mathbf{B} \to \mathbf{A}$,
	A is an object of $\mathbf{A}$,
we define the category	$(A{\downarrow}R)$ as follows:
An object of	$(A{\downarrow}R)$
is a pair	(C, η)
in which	$C : \mathbf{B}_0$
and	$\eta : \mathrm{Hom}_{\mathbf{A}}(A, RC)$;
so	$(C, \eta) : \Sigma C{:}\mathbf{B}_0.\, \mathrm{Hom}_{\mathbf{A}}(A, RC)$
and	$(A{\downarrow}R)_0 := \Sigma C{:}\mathbf{B}_0.\, \mathrm{Hom}_{\mathbf{A}}(A, RC)$.
A morphism	$f : (C, \eta) \to (D, g)$ in $(A{\downarrow}R)$
is an	$f : \mathrm{Hom}_{\mathbf{B}}(C, D)$ such that $Rf \circ \eta = g$,
or equivalently a pair	$(f, \langle\!\langle Rf \circ \eta = g \rangle\!\rangle)$;
we have	$(f, \langle\!\langle Rf \circ \eta = g \rangle\!\rangle) : \Sigma f{:}\, \mathrm{Hom}_{\mathbf{B}}(C, D).[\![Rf \circ \eta = g]\!]$,
so	$\mathrm{Hom}_{(A{\downarrow}R)}((C, \eta), (D, g)) :=$
	$\Sigma f{:}\, \mathrm{Hom}_{\mathbf{B}}(C, D).[\![Rf \circ \eta = g]\!]$.

The notations $\langle\!\langle P \rangle\!\rangle$ and $[\![P]\!]$ are non-standard. For any proposition P we denote by $[\![P]\!]$ the set of witnesses of P (see [HOTT, p.18]) and by $\langle\!\langle P \rangle\!\rangle$ a witness that P is true; formally, $\langle\!\langle P \rangle\!\rangle$ is a variable (with a long name!) whose type is $[\![P]\!]$, and $[\![P]\!]$ is a singleton when P is true and the empty set when P is false. A good way to remember this notation is that $[\![P]\!]$ looks like a box and $\langle\!\langle P \rangle\!\rangle$ looks like something that comes in that box.

This defines formally the first two components of the category $(A{\downarrow}R)$. Remember that a category $\mathbf{C}$ has seven components:

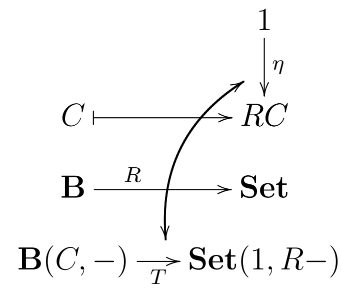
$$\mathbf{C} = (\mathbf{C}_0, \mathrm{Hom}_{\mathbf{C}}, \mathrm{id}_{\mathbf{C}}, \circ_{\mathbf{C}}; \mathsf{assoc}_{\mathbf{C}}, \mathsf{idL}_{\mathbf{C}}, \mathsf{idR}_{\mathbf{C}})$$

We are pretending that the other components of $(A{\downarrow}R)$ are "obvious" in the sense of Section 3.

7.2 The Yoneda Lemma

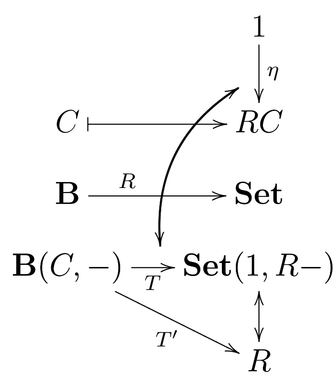
The formalization of $\mathsf{Y0}$ as a series of typings and definitions in Section 6.4 *suggests* that *some* operations from Type Theory that can be applied on the formalization side should be translatable to the diagram side; for example, substitution. This one clearly works: if we substitute $\mathbf{A}$ by $\mathbf{Set}$ and A by the set 1 we get this,

$$\mathsf{Y0}\begin{bmatrix}\mathbf{A} := \mathbf{Set} \\ A := 1\end{bmatrix} \quad =$$



For each $D \in \mathbf{B}$ we have a bijection $\mathbf{Set}(1, RD) \leftrightarrow RD$ — and we can use these bijections to build a natural isomorphism $\mathbf{Set}(1, R-) \leftrightarrow R$, that we will add to the diagram:

$$\mathsf{Y1} \quad :=$$



We can obtain T' from T and vice-versa by composing them with $\mathbf{Set}(1, R-) \leftrightarrow R$.

The diagram $\mathsf{Y1}$ "is" the Yoneda Lemma — but it doesn't have a single top-level arrow, so we can't apply the convention (CTL) to it, and we need to specify its "meaning" explicitly. The statement of the Yoneda Lemma is that there is a bijection

$$RC \leftrightarrow \mathrm{Hom}(\mathbf{B}(C, -), R);$$

Once we know that it is easy to see that the diagram $\mathsf{Y1}$ shows how we can build it by combining three bijections that we understand well:

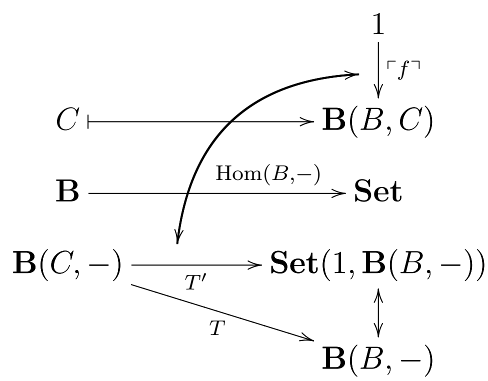
$$\begin{array}{l} RC \\ \leftrightarrow \mathrm{Hom}(1, RC) \\ \leftrightarrow \mathrm{Hom}(\mathbf{B}(C,-), \mathbf{Set}(1, R-)) \\ \leftrightarrow \mathrm{Hom}(\mathbf{B}(C,-), R) \end{array}$$

So $\mathsf{Y1}$ shows a way to build the bijection $RC \leftrightarrow \mathrm{Hom}(\mathbf{B}(C,-), R)$.

7.3 The Yoneda embedding

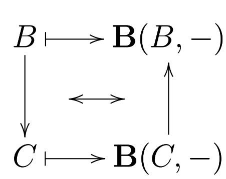
Let B be an object of $\mathbf{B}$. If we replace the functor $R : \mathbf{B} \to \mathbf{Set}$ in $\mathsf{Y1}$ by $\mathbf{B}(B,-)$ and do some other renamings we get this:

$$\mathsf{Y1}\begin{bmatrix} R := \mathbf{B}(B,-) \\ \eta := \ulcorner f \urcorner \\ T := T' \\ T' := T \end{bmatrix} :=$$



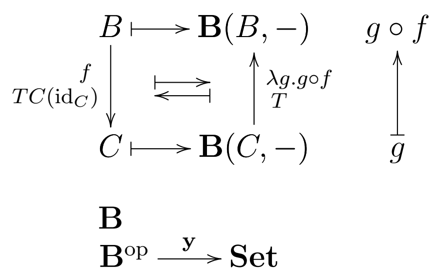
We can consider that the diagram above is a skeleton for the *proof* that there is a bijection between arrows $f : B \to C$ and natural transformations $T : \mathbf{B}(C,-) \to \mathbf{B}(B,-)$. The two directions of the bijection are easy to define, as $T_0 := \lambda D.\lambda g.g \circ f$ and $f := TC(\mathrm{id}_C)$, but the proof that the round trips $f \mapsto T \mapsto f$ and $T \mapsto f \mapsto T$ give back the original f and T are tricky, as we saw in Section 6.3.

Usually people draw a simple diagram that just *states* that the obvious map $\mathbf{B}(B,C) \to \mathrm{Hom}(\mathbf{B}(C,-), \mathbf{B}(B,-)$ is a bijection, somehow like this:



Compare with [Riehl, p.60]; note that our arrow in the middle of the square is a '$\leftrightarrow$'.

We can draw it with more details as:



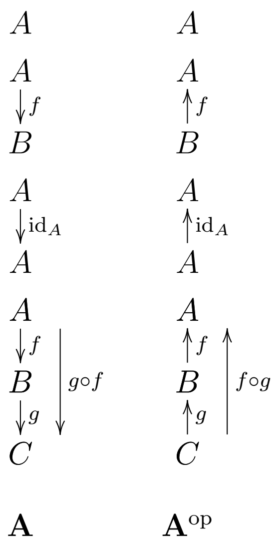
Note that it defines a *contravariant* functor $\mathbf{y} : \mathbf{B}^{\text{op}} \to \mathbf{Set}$ whose action on objects is $C \mapsto \mathbf{B}(C, -)$.

We consider that the morphism $f : B \to C$ in the diagram is inside $\mathbf{B}$, not inside $\mathbf{B}^{\text{op}}$. This is explained in the next section.

7.4 Opposite categories

Suppose that we have a diagram $A \overset{f}{\to} B \overset{g}{\to} C$ in a category $\mathbf{A}$. There are several different notations for the corresponding diagram in $\mathbf{A}^{\text{op}}$: for example, in [CWM, p.33] it would be written as $A \overset{f^{\text{op}}}{\leftarrow} B \overset{g^{\text{op}}}{\leftarrow} C$, while in [AT11, p.15] as $A \overset{f}{\leftarrow} B \overset{g}{\leftarrow} C$. The convention (COT) says that the notation in our diagrams should be as close as possible to the notation in the original text — so let's see how to support the notation in [AT11], that looks a bit harder than the one in [CWM].

We want to define a new category, $\mathbf{A}^{\text{op}}$, using tricks similar to the ones in Section 7.1, but now we can't pretend that the new composition is obvious. We will define $(\mathbf{A}^{\text{op}})_0$, $\text{Hom}_{\mathbf{A}^{\text{op}}}$, $\text{id}_{\mathbf{A}^{\text{op}}}$, and $\circ_{\mathbf{A}^{\text{op}}}$ without any textual explanations, with just the diagrams to convince the readed that our definitions are reasonable.



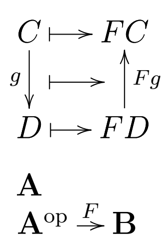
$$\mathbf{A}_0 =: (\mathbf{A}^{\mathrm{op}})_0$$

$$\mathrm{Hom}_{\mathbf{A}}(A, B) =: \mathrm{Hom}_{\mathbf{A}^{\mathrm{op}}}(B, A)$$

$$\mathrm{id}_{\mathbf{A}}(A) =: \mathrm{id}_{\mathbf{A}^{\mathrm{op}}}(A)$$

$$g \circ_{\mathbf{A}} f =: f \circ_{\mathbf{A}^{\mathrm{op}}} g$$

In the diagram below $F : \mathbf{A}^{\mathrm{op}} \to \mathbf{B}$ is a contravariant functor, and the $\mathbf{A}$ above $\mathbf{A}^{\mathrm{op}}$ indicates that $g : C \to D$ is a morphism of $\mathbf{A}$, not of $\mathbf{A}^{\mathrm{op}}$. I am not very happy with this trick but I haven't found a better alternative yet.



7.5 Universalness as something extra

We can consider that an universal arrow is an arrow $\eta : A \to RC$ with an extra property; I showed at the end of Section 4.2 how to think of that property as being just $\forall D.\forall g.\exists! f$, and how to treat that as an abbreviation for something bigger and more formal.

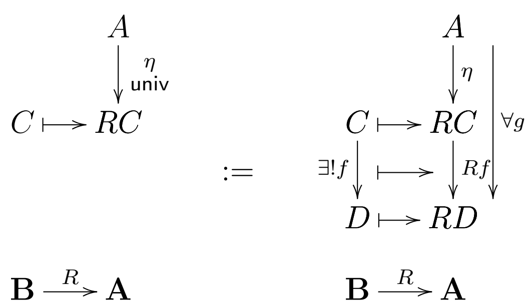
We can also treat a universal arrow as an arrow $\eta : A \to RC$ plus extra *structure* — this extra structure is an operation that returns for each D an inverse for the operation $g \mapsto Rg \circ \eta$. For more on properties and structure, see [BS07, p.15].

In any case this "universalness" can be treated as something extra, and a universal arrow can be expressed as:

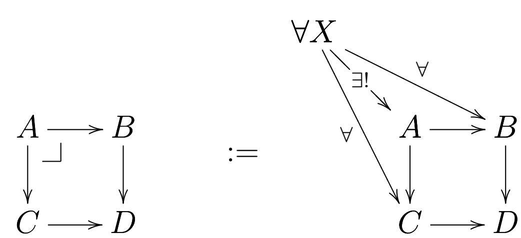
$$(\eta, \mathsf{univ}_\eta)$$

using dependent types.

Several of these "-ness"es have standard graphical representations: for example pullbackness is indicated by a '⌟', and monicness is indicated by a tail like this: '↣'. [FS90] defines lots of graphical representations for "-ness"es starting on its page 37. We will use an ':=' to define a new annotation that is an abbreviation for extra structure:

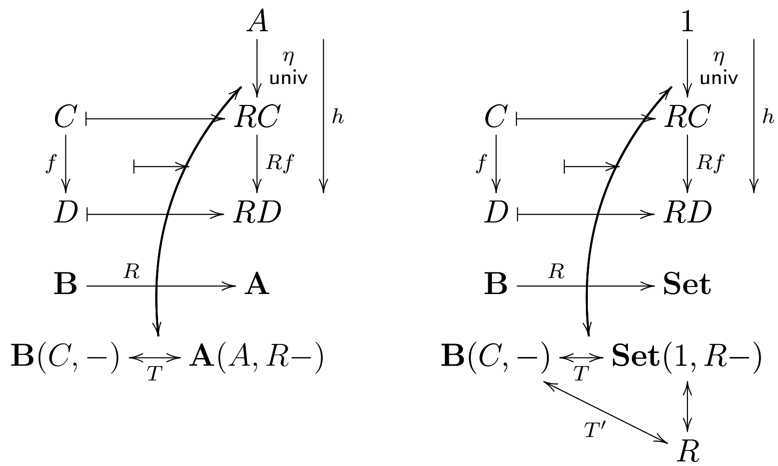


This is pullbackness:



7.6 Representable functors

It is easy to see that in Y0 the universality of η is equivalent to the natural-iso-ness of T; in Y1 the universality of η is equivalent to the natural-iso-ness of T, and this is equivalent to the natural-iso-ness of T'. The constructions should be evident from these diagrams:



The diagram at the right above can be seen as the missing diagram for Proposition 2 in [CWM, p.60], that says this (I've translated its letters to the ones I use):

> **Definition.** *Let* $\mathbf{B}$ *have small hom-sets. A representation of a functor* $R : \mathbf{B} \to \mathbf{Set}$ *is a pair* $\langle C, T' \rangle$, *with* C *an object of* $\mathbf{B}$ *and*
>
> $$T' : \mathbf{B}(C, -) \to R$$
>
> *a natural isomorphism. The object* C *is called the representing object. The functor* R *is said to be representable when such a representation exists.*
>
> Up to isomorphism, a representable functor is thus just a covariant hom-functor $\mathbf{B}(C, -)$. This notion can be related to universal arrows as follows.
>
> **Proposition 2.** Let 1 denote any one-point set and let $\mathbf{B}$ have small hom-sets. If $\langle C, \eta : 1 \to RC \rangle$ is a universal arrow from 1 to $R : \mathbf{B} \to \mathbf{Set}$, then the function T' which for each object D of $\mathbf{B}$ sends the arrow $f : C \to D$ to $(Rf)(\eta(*)) \in RD$ is a representation of R. Every representation of R is obtained in this way from exactly one such universal arrow.

The operations $T' \mapsto \eta$ and $\eta \mapsto T'$ can be defined as:

$$\begin{array}{rcl} \eta & : & 1 \to RC \\ T' & : & \mathbf{B}(C, -) \to R \\ \eta & := & \lambda * .(T'C(\mathrm{id}(C))) \\ T' & := & \lambda D.\lambda f.(Rf)(\eta(*)) \end{array}$$

7.7 An example of a representable functor

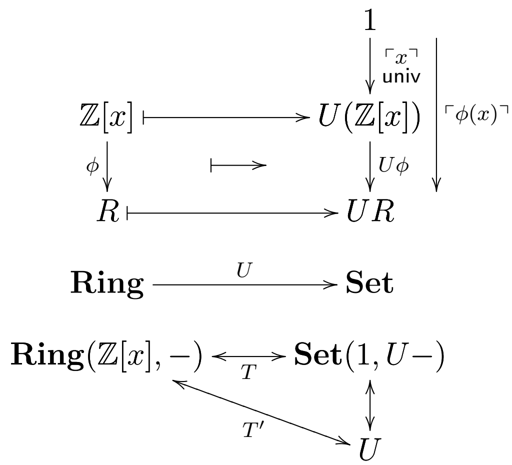
Emily Riehl gives two pages of examples of representable functors in [Riehl, pages 51–53]. Her example (iv) is:

> (iv) The functor U : **Ring** $\to$ **Set** is represented by the unital ring $\mathbb{Z}[x]$, the polynomial ring in one variable with integer coefficients. A unital ring homomorphism $Z[x] \to R$ is uniquely determined by the image of x; put another way, $\mathbb{Z}[x]$ is the *free unital ring on a single generator*.

She develops more this example in page 63, as:

> **Example 2.3.4.** Recall from Example 2.1.5(iv) that the forgetful functor U : **Ring** $\to$ **Set** is represented by the ring $Z[x]$. The universal element, which defines the natural isomorphism
>
> $$\mathbf{Ring}(Z[x], R) \cong UR,$$
>
> is the element $x \in \mathbb{Z}[x]$. As in the proof of the Yoneda lemma, the bijection above is implemented by evaluating a ring homomorphism $\phi : \mathbb{Z}[x] \to R$ at the element $x \in \mathbb{Z}[x]$ to obtain an element $\phi(x) \in R$.

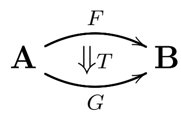
Here is the "missing diagram" for both excerpts:



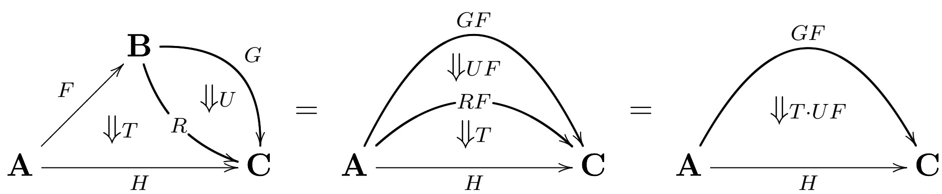
That diagram may be a good starting point to explain the Yoneda Lemma to "children".

7.8 The 2-category of categories

Natural transformations are often drawn as '$\Rightarrow$'s in the middle of "cells" whose walls are functors. If $F, G : \mathbf{A} \to \mathbf{B}$ are functors and $T : F \to G$ is natural transformation, then $\mathbf{A}, \mathbf{B}, F, G, T$ are drawn like this:

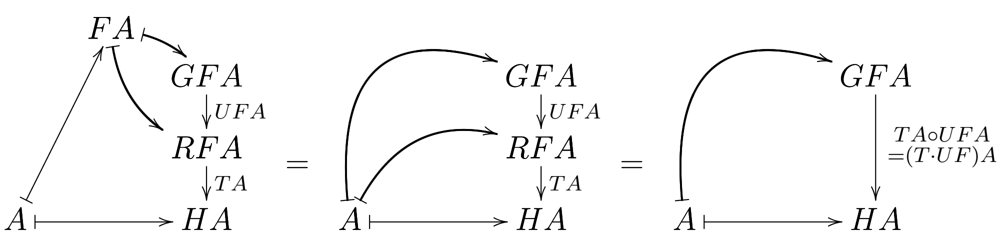


There are several ways to compose functors and natural transformations — see [Riehl, section 1.7] and [Pow90] for the details and the precise terminology. For example, in



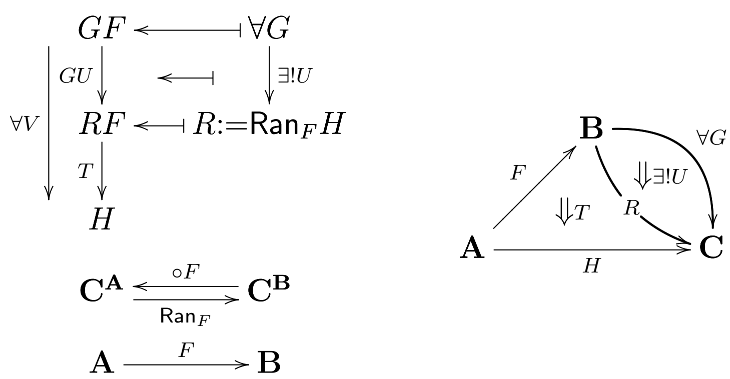
we used "whiskering" and then "vertical composition".

We can use internal views to lower the level of abstraction of the diagrams above. If we draw the images of an object $A \in \mathbf{A}$ by the functors and natural transformations we get:



7.9 Kan extensions

Kan extensions are usually drawn using 2-cells ([Riehl, definition 6.1.1]), but they can also be drawn as adjunctions ([Riehl, proposition 6.1.5], [CWM, section X.3]). Let's see how to draw them in both ways at the same time in a way that makes the translation clear. Here is the diagram:



We will consider right Kan extensions only.

Fix $F : \mathbf{A} \to \mathbf{B}$ and a category $\mathbf{C}$. We have a functor $(\circ F) : \mathbf{C}^{\mathbf{B}} \to \mathbf{C}^{\mathbf{A}}$. Suppose that it has a right adjoint, $(\circ F) \dashv \mathsf{Ran}_F$. For each natural transformation $H : \mathbf{A} \to \mathbf{C}$ its image by Ran_f, $R := \mathsf{Ran}_F H$, is a natural transformation $R : \mathbf{B} \to \mathbf{C}$. We have:

$$\begin{array}{rcl} \mathrm{Hom}_{\mathbf{C}^{\mathbf{A}}}(GF, H) & \cong & \mathrm{Hom}_{\mathbf{C}^{\mathbf{B}}}(G, R) \\ \mathrm{Hom}_{\mathbf{C}^{\mathbf{A}}}(- \circ F, H) & \cong & \mathrm{Hom}_{\mathbf{C}^{\mathbf{B}}}(-, \mathsf{Ran}_F H), \end{array}$$

and if substitute $[- := \mathsf{Ran}_F H]$ and we transpose $\mathrm{id}_{\mathsf{Ran}_F H}$ to the left we obtain a morphism $T : RF \to H$. The pair (R, H) obeys a certain universal property, that we will call "Ran-ness":

$$\forall G : \mathbf{B} \to \mathbf{C}.\ \forall V : GF \to H.\ \exists! U : G \to R.\ (T \cdot UF) = V.$$

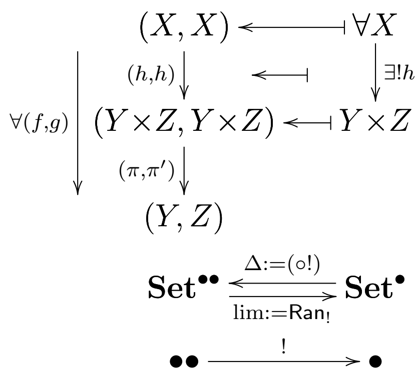
The usual way of defining right Kan extensions is by starting with the functors $F : \mathbf{A} \to \mathbf{B}$ and $H : \mathbf{A} \to \mathbf{C}$ and then saying that a pair (R, T) is a right Kan extension of H along F iff it obeys Ran-ness; the functor Ran_F and the adjunction come later. See [Riehl], section 6.1.

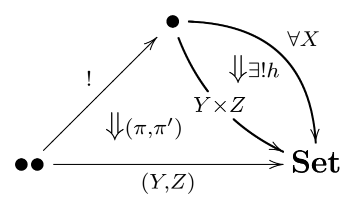
Note that we don't draw the '$\forall V : GF \to H$' in the right half of the diagram — it would overwrite the rest.

7.10 All concepts are Kan extensions

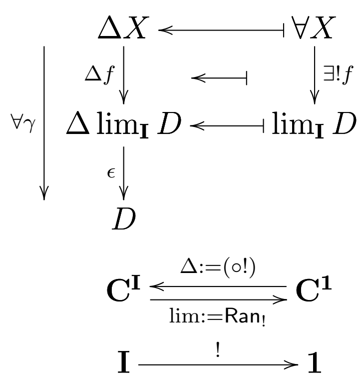
Both [CWM] and [Riehl] have sections called "All concepts are Kan extensions" — section X.7 in [CWM] and 6.5 in [Riehl]. Now that we have a favorite way of drawing right Kan extensions we can use it to draw diagrams for 1) binary products in **Set** are right Kan extensions, 2) limits are right Kan extensions and 3) left adjoints are right Kan extensions.

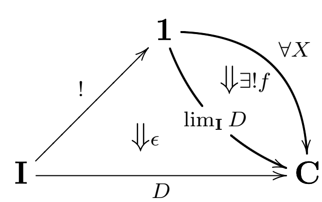
1. Let •• be the discrete category with two objects, • be the discrete category with one object, and $! : {\bullet\bullet} \to \bullet$ be the unique functor from •• to •. Then:





2. Let **I** be a finite index category — for example, $\mathbf{I} = \left(\begin{smallmatrix} & 1 \\ & \downarrow \\ 2 \to & 3 \end{smallmatrix}\right)$ — and let **C** be a category with finite limits. A functor $D : \mathbf{I} \to \mathbf{C}$ is a diagram of shape **I** in **C**. Let's denote by **1** the discrete category with a single object — the name '**1**' is more standard than '•'. Then:

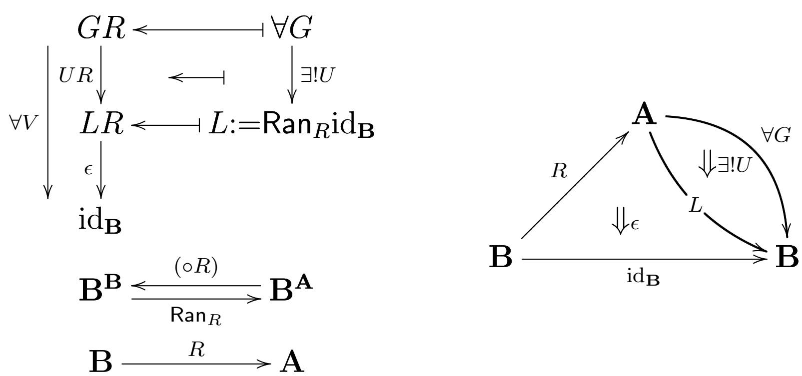




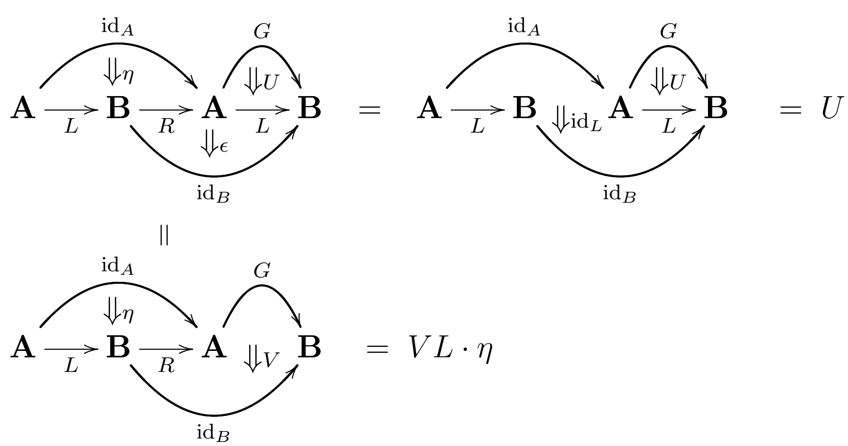
3. Left adjoints are right Kan extensions. If

$$\mathbf{B} \underset{R}{\overset{L}{\leftrightarrows}} \mathbf{A}$$

is an adjunction, then (L, ϵ) is a right Kan extension of $\mathrm{id}_\mathbf{B}$ along R. In a more compact notation, $L := \mathsf{Ran}_R \mathrm{id}_\mathbf{B}$ — but in this case we only know the action of Ran_R on the object $\mathrm{id}_\mathbf{B}$, and we don't know if this Ran_R can be extended to a "real" functor whose domain is the whole of $\mathbf{B}^\mathbf{B}$. The diagram is:



To show that this works we have to prove that $\forall V.\exists! U.(\epsilon \cdot UR = V)$. We will do that by "inverting the equation $\epsilon \cdot UR = V$":



The solution in $U := VL \cdot G\eta$.

7.11 A formula for Kan extensions

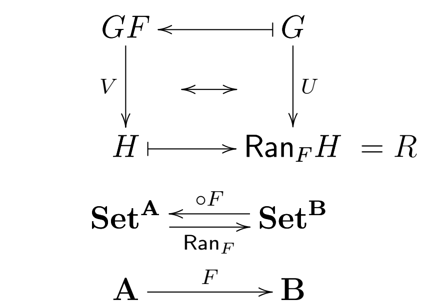
The sections X.3 of [CWM] and 6.2 of [Riehl] discuss a formula for calculating Kan extensions, that defines $\mathsf{Ran}_F H$ as the functor whose action on objects is:

$$B \mapsto \mathrm{Lim}(B{\downarrow}F \xrightarrow{\pi} \mathbf{A} \xrightarrow{H} \mathbf{Set}),$$

and its action on morphisms is "obvious" in the sense of Section 3. I found this formula totally impossible to understand until I finally found a way to visualize what it "meant".

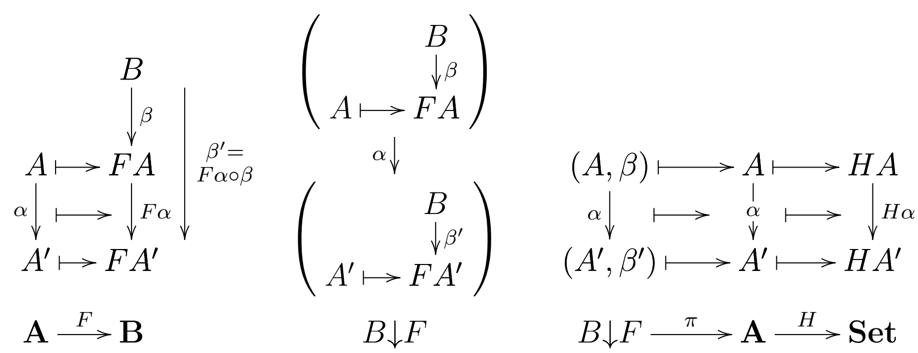
For each object $B \in \mathbf{B}$ the functor $B{\downarrow}F \xrightarrow{H\circ\pi} \mathbf{Set}$ can be regarded as a diagram in **Set** whose shape is the shape of the comma category $B{\downarrow}F$. If **A** and **B** are finite preorder categories and F is an inclusion then $B{\downarrow}F$ can "inherit its shape" from **A**; inclusions of preorders are "toy examples" "for children", but they give us some intuition to start with, and they can help us understand the formal version that can handle more general cases.

These are the diagrams for Ran_F as a right adjoint — note that we use **Set** instead of **C** to make things less abstract,



$$\begin{array}{l} RB = \\ (\mathsf{Ran}_F H)B = \\ \mathrm{Lim}(B{\downarrow}F \xrightarrow{\pi} \mathbf{A} \xrightarrow{H} \mathbf{Set}) \end{array}$$

and here are some diagrams to help us understand the comma category $B{\downarrow}F$ — in the compact notation its objects have names like (A, β), but in the more visual notation they are L-shaped diagrams:



Let's see an example.

If $\mathbf{A} \overset{F}{\to} \mathbf{B}$ is the inclusion $\begin{pmatrix} & 2 \\ & \downarrow \\ 5 \longrightarrow & 6 \end{pmatrix} \to \begin{pmatrix} 1' \to 2' \\ \downarrow \quad \downarrow \\ 3' \to 4' \\ \downarrow \quad \downarrow \\ 5' \to 6' \end{pmatrix}$,

then $1'{\downarrow}F = \begin{pmatrix} & \left(\begin{smallmatrix} & 1' \\ 2 & F2 \end{smallmatrix}\right) \\ & \downarrow \\ \left(\begin{smallmatrix} & 1' \\ 5 & F5 \end{smallmatrix}\right) \longrightarrow & \left(\begin{smallmatrix} & 1' \\ 6 & F6 \end{smallmatrix}\right) \end{pmatrix}$ and $3'{\downarrow}F = \begin{pmatrix} & \\ & \\ \left(\begin{smallmatrix} & 3' \\ 5 & F5 \end{smallmatrix}\right) \longrightarrow & \left(\begin{smallmatrix} & 3' \\ 6 & F6 \end{smallmatrix}\right) \end{pmatrix}$,

and $(1'{\downarrow}F \overset{H\circ\pi}{\to} \mathbf{Set}) = \begin{pmatrix} & H_2 \\ & \downarrow \\ H_5 \longrightarrow & H_6 \end{pmatrix}$ and $(3'{\downarrow}F \overset{H\circ\pi}{\to} \mathbf{Set}) = \begin{pmatrix} & \\ & \\ H_5 \longrightarrow & H_6 \end{pmatrix}$;

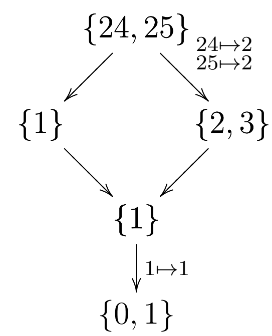
so $R(1') = \mathrm{Lim}(1'{\downarrow}F \overset{H\circ\pi}{\to} \mathbf{Set}) = H_2 \times_{H_6} H_5$,
and $R(3') = \mathrm{Lim}(3'{\downarrow}F \overset{H\circ\pi}{\to} \mathbf{Set}) = H_5$.
We can follow the same pattern to calculate $R(2')$, $R(4')$, $R(5')$, $R(6')$.
The square of the adjunction becomes this, in this particular case:

$$\begin{array}{ccc} GF & \longleftarrow\!\!| & G \\ \downarrow & \longleftrightarrow & \downarrow \\ H & \longmapsto & \mathsf{Ran}_F H \; = R \end{array} \qquad \begin{array}{ccc} \begin{pmatrix} & G_2 \\ & \downarrow \\ G_5 \longrightarrow & G_6 \end{pmatrix} & \longleftarrow\!\!| & \begin{pmatrix} G_1 \longrightarrow G_2 \\ \downarrow \qquad \downarrow \\ G_3 \longrightarrow G_4 \\ \downarrow \qquad \downarrow \\ G_5 \longrightarrow G_6 \end{pmatrix} \\ \downarrow & \longleftrightarrow & \downarrow \\ \begin{pmatrix} & H_2 \\ & \downarrow \\ H_5 \longrightarrow & H_6 \end{pmatrix} & \longmapsto & \begin{pmatrix} H_2 \times_{H_6} H_5 \longrightarrow H_2 \\ \downarrow \qquad \downarrow \\ H_5 \longrightarrow H_6 \\ \downarrow \qquad \downarrow \\ H_5 \longrightarrow H_6 \end{pmatrix} \end{array}$$

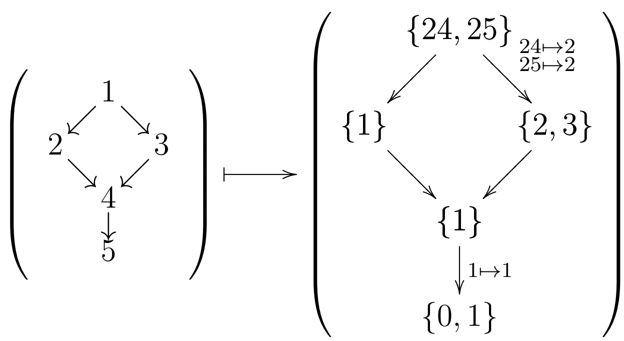
7.12 Functors as objects

One way to treat a diagram in **Set** like this

$F \quad :=$



as a functor is to think that that diagram is an abbreviation — it is just the upper-right part of a diagram like this,



$$\mathbf{K} \xrightarrow{\quad F \quad} \mathbf{Set}$$

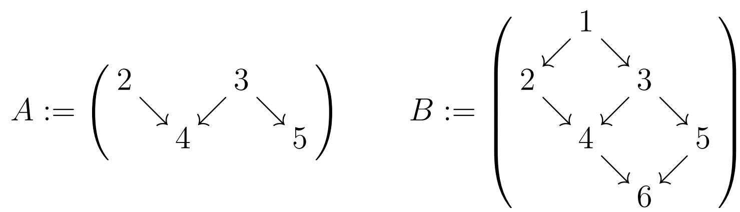
where we add the extra hint that the index category **K** is exactly the kite-shaped preorder category drawn above the "**K**".

The convention (CFSh) says that the image by a functor of a diagram is a diagram with the same shape, so according to that convention we have $F(1) = \{24, 25\}$, $F(4 \to 5) = (\{1\} \overset{1 \mapsto 1}{\to} \{0, 1\})$, and so on; so the upper right part of the diagram above *defines* F.

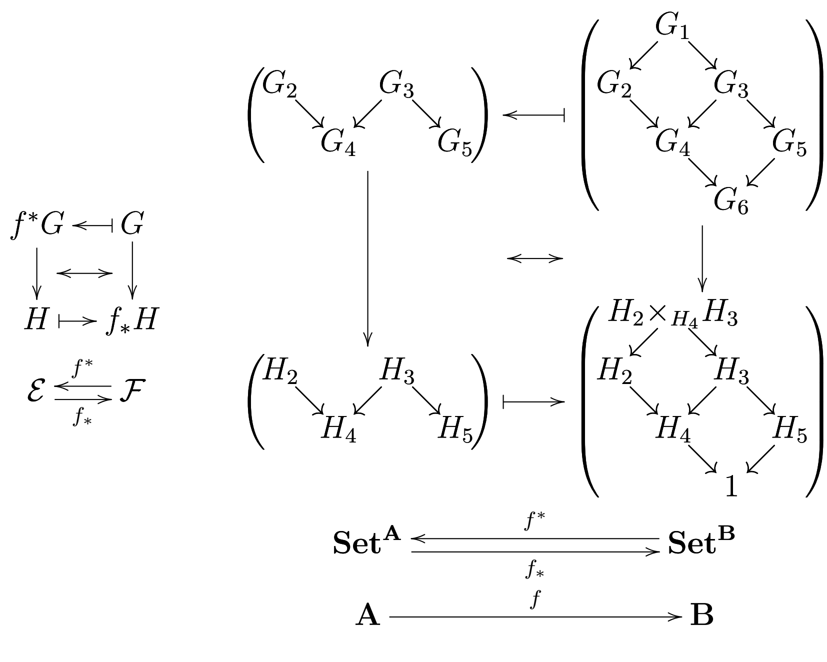
Note that the single '$\mapsto$' above the $\mathbf{K} \overset{F}{\to} \mathbf{Set}$ stands for several '$\mapsto$'s, one for each object and one for each morphism, and note that F is an object of $\mathbf{Set}^{\mathbf{K}}$.

7.13 Geometric morphisms for children

Let **A** and **B** be these preorder categories, and let $f : \mathbf{A} \to \mathbf{B}$ be the inclusion functor from **A** to **B**:



The left half of the diagram below is the standard definition of a geometric morphism f from a topos $\mathcal{E}$ to a topos $\mathcal{F}$. A geometric morphism $f : \mathcal{E} \to \mathcal{F}$ is actually an adjunction $f^* \dashv f_*$ plus the guarantee that $f^* : \mathcal{E} \leftarrow \mathcal{F}$ preserves limits, which is a condition slightly weaker than requiring that f^* has a left adjoint. When that left adjoint exists it is denoted by $f^!$, and we say that $f^! \dashv f^* \dashv f_*$ is an *essential geometric morphism*. The only non-standard thing about the diagram at the left below is that is contains an internal view of the adjunction $f^* \dashv f_*$.



The right half of the diagram is a particular case of the left half. Its lower line, $\mathbf{A} \xrightarrow{f} \mathbf{B}$, does not exist in the left half. The inclusion functor f induces adjunctions $f^! \dashv f^* \dashv f_*$ as this,

$$\begin{array}{ccc} \mathbf{Set}^{\mathbf{A}} & \begin{array}{c} \xrightarrow{f^!} \\ \xleftarrow{f^*} \\ \xrightarrow[f_*]{} \end{array} & \mathbf{Set}^{\mathbf{B}} \\ \\ \mathbf{A} & \xrightarrow{f} & \mathbf{B} \end{array}$$

where f^* is easy to define and $f^!$ and f_* not so much — the standard way to define $f^!$ and f_* is by Kan extensions.

The big square in the upper part of the diagram is an internal view of the adjunction $f^* \dashv f_*$, with the functors f^*G, G, H, and f_*H being displayed as their internal views. We can choose the sets $G_1, \ldots, G_6$ and the morphisms between them arbitrarily, so this is an internal view of an arbitrary functor $G : \mathbf{B} \to \mathbf{Set}$; and the same for H.

The arrow $f^*G \hookleftarrow G$ can be read as a definition for the action of f^* on objects — it just erases some parts of the diagram — and the arrow $H \mapsto f_*H$ can be read as a definition for the action of f_* on objects — f_* "reconstructs" H_1 and H_6 in a certain natural way. It is easy to reconstruct the actions of f^* and f_* on morphisms from just what is shown, and to reconstruct the two directions of the bijection.

The big diagram above can be used 1) to convince people that are not hardcore toposophers that this diagrammatic language can make some difficult categorical concepts more accessible, and 2) as a starting point to generate diagrams "for children" for several parts of the Elephant, and even to prove new theorems on toposes. For more on (1), see [OL18] and [Och18]; for (2), see [MDE].

7.14 Reading the Elephant

In Section 5.5 we saw a strategy for helping (beginner) students to read a difficult text on CT: we start with diagrams for the most important concepts, in both a general case "for adults" and a well-chosen particular case "for children", we give them exercises to make sure that they understand the constructions in the case "for children", we give them a few more exercises to make sure that they understand the general case, we ask them to read excerpts from a standard textbook in a version where the letters were changed to match the diagrams, and then we ask them to work on the original version of these excerpts with the original notation, and on some other parts of the same chapter... this can be done for the Elephant too — here are the parts that are more relevant for our diagrams on geometric morphisms, with the notation adjusted:

> **Definition 4.1.1.** (a) Let $\mathcal{F}$ and $\mathcal{E}$ be toposes. A geometric morphism $f : \mathcal{E} \to \mathcal{F}$ consists of a pair of functors $f_* : \mathcal{E} \to \mathcal{F}$ (the direct image of f) and $f^* : \mathcal{F} \to \mathcal{E}$ (the inverse image of f) together with an adjunction $(f^* \dashv f_*)$, such that f^* is cartesian (i.e. preserves finite limits).
>
> (...)
>
> **Example 4.1.4.** Let $f : \mathbf{A} \to \mathbf{B}$ be a functor between small categories. Then composition with f defines a functor $f^* : \mathbf{Set}^{\mathbf{B}} \to \mathbf{Set}^{\mathbf{A}}$, which has adjoints on both sides, the left and right *Kan extensions* along f: for example, the right Kan extension Ran_f sends a functor $H : \mathbf{Set}^{\mathbf{A}}$ to the functor whose value at an object B of $\mathbf{B}$ is the limit of the diagram
>
> $$(B \downarrow f) \xrightarrow{U} \mathbf{A} \xrightarrow{H} \mathbf{Set}$$
>
> (here $(B \downarrow f)$ is the comma category whose objects are pairs (A, ϕ) with $\phi : B \to fA$ in $\mathbf{B}$, and U is the forgetful functor from this category to $\mathbf{A}$). Thus f^* is the inverse image of a geometric morphism $\mathbf{Set}^{\mathbf{A}} \to \mathbf{Set}^{\mathbf{B}}$, whose direct image is Ran_f.
>
> (...)
>
> We note that the geometric morphisms which arise as in 4.1.4, though not as special as those of 4.1.2, still have the property that their inverse image functors have left adjoints as well as right adjoints. We call a geometric morphism f *essential* if it has this property; we normally write $f_!$ for the left adjoint of f^*. With the aid of this notion, we can prove a partial converse to 4.1.4:
>
> **Lemma 4.1.5.** Let $\mathbf{A}$ and $\mathbf{B}$ be small categories such that $\mathcal{B}$ is Cauchy-complete (cf. 1.1.10). Then every essential geometric morphism $f : \mathbf{Set}^{\mathbf{A}} \to \mathbf{Set}^{\mathbf{B}}$ is induced as in 4.1.4 by a functor $\mathbf{A} \to \mathbf{B}$.
>
> (...)

> **Proposition 4.2.8.** With the notation established above, the counit $h^*h_* \to 1$ is an isomorphism.
>
> (...)
>
> A geometric morphism h satisfying the condition that the counit $h^*h_* \to 1$ is an isomorphism, or the equivalent condition that h_* is full and faithful, is called an *inclusion* (though some authors prefer the term *embedding*). We shall study inclusions in greater detail in the next three sections; for the present, we digress briefly to note an alternative characterization of them:

The really interesting part would be to show that the unit η of the adjunction $f^* \dashv f_*$ "is" a sheafification functor, and that the geometric morphism for children of the diagram yields an example of sheaf... but that would need lots of different fragments from several different sections of the book.

8 How to name this diagrammatic language

I don't have any idea!...

It can be used to produce missing diagrams, and sometimes these missing diagrams are skeletons. We can use it to work in two styles in parallel, "for adults" and "for children"... maybe something like "Missing Skeletons for Children"?

Suggestions welcome.

9 Why "my conventions"?

I learned CT as an autodidact in a totally disorganized way. In the first years I just read, or rather *tried* to read, everything that was available in my university's library, trying to locate the parts that could be useful to my main interest at that time, that was Non-Standard Analysis and how to do something similar to NSA but using filter-powers instead of ultrapowers...

It was only after that that I realized that I had to learn how to *write*. I remember one time spending a whole evening on an exercise of the beginning of [LS86] that says just "prove that for categories $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$ we have $\mathbf{A}^{\mathbf{B}\times\mathbf{C}} \cong (\mathbf{A}^{\mathbf{B}})^{\mathbf{C}}$" — the full proof had lots of parts, and I saw that I didn't know how to organize them in a neat way... also, the proofs given in books and articles just state the main parts and pretend that the rest is obvious, and in the case of $\mathbf{A}^{\mathbf{B}\times\mathbf{C}} \cong (\mathbf{A}^{\mathbf{B}})^{\mathbf{C}}$ there were no "main parts", so I had to learn how to write down a proof in full, and this was a new style to me...

Even now, many years after that, I still have the sensation that I had to improvise practically everything in my ways — both the "algebraic" way and the "diagrammatic" way — of writing categorical proofs, and that I still don't know even a tiny fraction of the techniques for writing that people learn when they take CT courses and they have opportunities to discuss exercises with other students and with TAs and more senior people...

The "my conventions" in the title of this text, and my use of the first person everywhere, are a way to stress that I still don't know enough about other people's private languages for CT, and that this is an attempt to gain access to other private languages, diagrammatic or not... I am especially interested in how people write when they turn their level-of-detail knob to a very high position.

10 Related and unrelated work

The diagrammatic language that I described here seems to be unrelated to the ones in [CK17] and [Coe11] — that describe *lots* of diagrammatic languages — and also unrelated to [Mar14]. We lower the level of abstraction — see for example Section 7.8 — while they (usually) raise it.

I've taken an approach that is the opposite of [CW01] and [Các04]. Cáccamo and Winskel define a derivation system that can only construct functors, natural transformations, etc, that obey the expected naturality conditions, while we allow some kinds of sloppinesses, like constructing something that looks like a functor and pretending that it is a functor when it may not be. When I started working on this diagrammatic language I had a companion derivation system for it; [IDARCT, Section 14] mentions it briefly, but it doesn't show the introduction rules that create (proto)functors and (proto)natural transformations and that allow being sloppy ("in the syntactical world").

Some of my excuses for allowing one to pretend that a functor is a functor and leaving the verification to a second stage come from [Che04]. I learned a *lot* on how mathematicians use intuition and diagrams from [Krö07] — [Krö18] is a great summary — and [Cor04], and they have helped me to identify which characteristics of my diagrammatic language are very unusual and may be new, and that deserve to be presented in detail.

Many of the first ideas for my diagrammatic language appeared when I was reading [See83], [See84], [See87], [Jac99], and [BCS06] and trying to draw the "missing diagrams" in those papers in both the original notation and in the "archetypal case" ([IDARCT, Section 16]).

11 What next?

At this point I think that it is more interesting to "implement" more categorical definitions and proofs in this diagrammatic language than to try to formalize it completely or try to prove meta-theorems about it. I am doing that by (re)reading parts of several papers and articles and drawing the missing diagrams in them; for details and links, see:

`http://angg.twu.net/math-b.html#favorite-conventions`

Besides this, here's what I've planned for the next steps. Most of them can be done in parallel.

1. Now there are several very good books on CT for beginners with lots of diagrams — for example [FS19], [Perrone], and [Mil20]. I want to try do draw the "missing diagrams" for some of their sections, show them to some people, and see if they find them useful.

2. I need to learn more Idris and Idris-ct — and then 1) draw the missing diagrams for some of the modules in the Idris-ct sources (as a visual guide for the names of the data structures and their fields), 2) implement some of my diagrams on Idris-ct; a column with Idris-ct code would be a nice addition to, for example, Section 6.4.

3. The paper [PH2] that I uploaded to Arxiv is a kind of "Sheaves for Children", and some philosopher friends of mine who study Alain Badiou — who uses toposes and sheaves in books like [Bad09] and [Bad14] — expressed a lot of interest in it... the first six sections of [PH2] are impeccable (I think!) but the last ones, that are the ones that involve categories, were written in a hurry. I need to rewrite them using techniques like the ones in Section 5.5 to turn them into something like a "Let's read some sections of [Elephant] and [Riehl] — an illustrated

guide"... until I finish that I can't advertise [PH2], I am too embarassed by its last sections.

References

[AT11] S. Abramsky and N. Tzevelekos. "Introduction to Categories and Categorical Logic". https://arxiv.org/pdf/1102.1313.pdf. 2011.

[Awo06] S. Awodey. *Category Theory*. Oxford, 2006.

[Bad09] A. Badiou. *Logics of Worlds - Being and Event, 2*. Continuum, 2009.

[Bad14] A. Badiou. *Mathematics of The Transcendental*. Bloomsbury, 2014.

[BCS06] R. Blute, R. Cockett, and R. A. G. Seely. "Differential Categories". In: *Mathematical Structures in Computer Science* 16 (2006). http://www.math.mcgill.ca/rags/difftl/difftl.pdf, pp. 1049–1083.

[Bra17] E. Brady. *Type-Driven Development With Idris*. Manning, 2017.

[BS07] J. Baez and M. Shulman. "Lectures on n-Categories and Cohomology". https://arxiv.org/pdf/math/0608420.pdf. 2007.

[Các04] M. J. Cáccamo. "A Formal Calculus for Categories". http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.1.7460. PhD thesis. Aarhus, 2004.

[Che04] E. Cheng. "Mathematics, Morally". http://eugeniacheng.com/wp-content/uploads/2017/02/cheng-morality.pdf. 2004.

[CK17] B. Coecke and A. Kissinger. *Picturing Quantum Processes. A First Course in Quantum Theory and Diagrammatic Reasoning*. http://www.cambridge.org/gb/pqp. Cambridge, 2017.

[Coe11] B. Coecke, ed. *New Structures for Physics*. https://www.springer.com/br/book/9783642128202. Springer, 2011.

[Cor04] D. Corfield. *Towards a Philosophy of Real Mathematics*. Cambridge, 2004.

[CW01] M. J. Cáccamo and G. Winskel. "A Higher-Order Calculus for Categories". https://www.brics.dk/RS/01/27/BRICS-RS-01-27.pdf. 2001.

[CWM] S. Mac Lane. *Categories for the Working Mathematician (2nd ed.)* Springer, 1997.

[Elephant] P. T. Johnstone. *Sketches of an Elephant: A Topos Theory Compendium*. Vol. 1. Oxford, 2002.

[ES52] S. Eilenberg and N. Steenrod. *Foundations of algebraic topology.* Princeton, 1952.

[Freyd76] P. Freyd. “Properties Invariant within Equivalence Types of Categories”. In: *Algebra, Topology and Category Theory: A Collection of Papers in Honour of Samuel Eilenberg*. Ed. by A. Heller and M. Tierney. http://angg.twu.net/Freyd76.html. Academic Press, 1976, pp. 55–61.

[FS19] B. Fong and D. I. Spivak. *Seven Sketches in Compositionality: An Invitation to Applied Category Theory*. https://arxiv.org/pdf/1803.05316.pdf. Cambridge, 2019.

[FS90] P. Freyd and A. Scedrov. *Categories, Allegories*. North-Holland, 1990.

[HOTT] The Univalent Foundations Program. *Homotopy Type Theory: Univalent Foundations of Mathematics*. http://saunders.phil.cmu.edu/book/hott-online.pdf. Institute for Advanced Study, 2013.

[IDARCT] E. Ochs. “Internal Diagrams and Archetypal Reasoning in Category Theory”. In: *Logica Universalis* 7.3 (Sept. 2013). http://angg.twu.net/math-b.html#idarct, pp. 291–321.

[IdrisCT] F. Genovese et al. *idris-ct: A Library to do Category Theory in Idris*. 2019. arXiv: 1912.06191 [cs.LO].

[Jac99] B. Jacobs. *Categorical Logic and Type Theory*. Studies in Logic and the Foundations of Mathematics 141. North-Holland, Elsevier, 1999.

[Jam01] M. Jamnik. *Mathematical Reasoning with Diagrams: From Intuition to Automation*. CSLI, 2001.

[Krö07] R. Krömer. *Tool and Object: A History and Philosophy of Category Theory*. https://www.springer.com/gp/book/9783764375232. Birkhäuser, 2007.

[Krö18] R. Krömer. “Category theory and its foundations: the role of diagrams and other “intuitive” material”. Slides for his keynote talk at the UniLog 2018. http://angg.twu.net/logic-for-children-2018/ralf_kroemer__slides.pdf. 2018.

[LR03] W. Lawvere and R. Rosebrugh. *Sets for Mathematics*. Cambridge, 2003.

[LS86] J. Lambek and P. Scott. *Introduction to Higher-Order Categorical Logic*. Cambridge, 1986.

[Mar14] D. Marsden. “Category Theory Using String Diagrams”. https://arxiv.org/pdf/1401.7220.pdf. 2014.

[MDE] E. Ochs. “On Some Missing Diagrams in The Elephant”. http://angg.twu.net/math-b.html#missing-diagrams-elephant. 2019.

[Mil20] B. Milewski. “Category Theory for Programmers, OCaml Edition”. https://github.com/hmemcpy/milewski-ctfp-pdf/releases/download/v1.4.0-rc1/category-theory-for-programmers-ocaml.pdf. 2020.

[Och18] E. Ochs. “Visualizing Geometric Morphisms”. http://angg.twu.net/LATEX/2018vichy-vgms-slides.pdf. 2018.

[Och19] E. Ochs. “How to almost teach Intuitionistic Logic to Discrete Mathematics Students”. http://angg.twu.net/LATEX/2019logicday.pdf. 2019.

[Och20] E. Ochs. “What kinds of knowledge do we gain by doing Category Theory in several levels of abstraction in parallel?” http://angg.twu.net/math-b.html#2020-tallinn. 2020.

[OL18] E. Ochs and F. Lucatelli. “Logic for Children - Workshop at UniLog 2018 (Vichy) - unofficial homepage”. http://angg.twu.net/logic-for-children-2018.html. 2018.

[Penrose] K. Ye et al. “Penrose: From Mathematical Notation to Beautiful Diagrams”. In: http://penrose.ink/media/Penrose_SIGGRAPH2020.pdf. 2020.

[Perrone] P. Perrone. “Notes on Category Theory with examples from basic mathematics”. https://arxiv.org/abs/1912.10642. 2020.

[PH1] E. Ochs. "Planar Heyting Algebras for Children". In: *South American Journal of Logic* 5.1 (2019). http://angg.twu.net/math-b.html#zhas-for-children-2, pp. 125–164.

[PH2] E. Ochs. "Planar Heyting Algebras for Children 2: Local J-Operators, Slashings, and Nuclei". http://angg.twu.net/math-b.html#zhas-for-children-2. 2021.

[Pow90] A. J. Power. "A 2-Categorical Pasting Theorem". In: *Journal of Algebra* 129.2 (1990). https://core.ac.uk/download/pdf/81929927.pdf, pp. 439–445.

[Riehl] E. Riehl. *Category Theory in Context*. http://www.math.jhu.edu/~eriehl/context.pdf. Dover, 2016.

[See83] R. A. G. Seely. "Hyperdoctrines, natural deduction, and the Beck condition". In: *Zeitschrift f. math. Logik und Grundlagen d. Math.* 29 (1983). http://www.math.mcgill.ca/rags/ZML/ZML.PDF, pp. 505–542.

[See84] R. A. G. Seely. "Locally Cartesian closed categories and type theory". In: *Math. Proc. Cambridge Philos. Soc.* 95.1 (1984). http://www.math.mcgill.ca/rags/LCCC/LCCC.pdf, pp. 33–48.

[See87] R. A. G. Seely. "Categorical Semantics for Higher Order Polymorphic Lambda Calculus". In: *Journal of Symbolic Logic* 52.4 (1987). http://www.math.mcgill.ca/rags/JSL/PLC.pdf, pp. 969–988.

[SICP] H. Abelson and G.J. Sussman. *Structure and Interpretation of Computer Programs, 2nd ed.* https://web.mit.edu/alexmv/6.037/sicp.pdf. MIT, 1996.