

# Summary of the things that I learned

Arnoud van der Leer



## CHAPTER 1

# Lessons about coq and unimath

When writing coq code, make sure you understand why a proof should work, instead of blindly unfolding and applying Lemmas. That improves the overall quality of the proofs.

A proof closed with `Qed` is opaque, whereas a proof that closes with `Defined` is transparent (i.e. is remembered can be unfolded). Which one is the right one requires some thought.

Only use `destruct` in opaque proofs.

Path induction (or induction on proofs of equality) helps a lot when proving something about a `transportf`.

### 1. Default API for any object

- `object_data`: A definition for the data of the object;
- `make_object_data`: A function to make the object data out of its parts;
- (Coercions from `object_data` to some of its parts;)
- (Explicit functions to access the constituents;)
- `is_object`: A definition for the properties of the object;
- `make_is_object`: A function to make the properties part of the object;
- `make_object`: A function to make the object out of its data and property components;
- `object`:
- (A coercion from the object to its data;)
- (`object_has_property`: Explicit functions to access the properties;)
- `is_object_isaprop`: A lemma that the properties form a proposition;
- `object_eq`: A lemma about sufficient (and necessary) conditions for two terms of type `object` to be equal (usually some conditions on the constituents of `object_data`);

### 2. Cleaning up code

It is okay to simplify proofs as much as possible. When one wants to step through a proof, they can add `simpls` to their own liking.

`cbn` is stronger than `simpl`, so it can be good to try and replace `cbns` with `simpls` when cleaning up code.

### 3. Things I was not able to find on my own

```
isweq_iso  
weqdnicoproduct
```

#### 4. Things I would have liked to know at the start

The fact that if a rewrite does not succeed because a lemma has a slightly different (folded or unfolded) description, we can just force its form like `rewrite (lemma : form1 = form2)`.

Setting just one implicit argument like `action (n := 5)`.

The fact that instead of finishing a (sub)proof with `[tactic]. apply idpath.`, one can finish it with `now [tactic]..`

#### 5. Things that I would like to have a standard for

- Naming (of folders, files, terms): what abbreviations are allowed? In what order do words appear? How are words separated? How are names capitalized?
- Documentation headers in files.
- In which cases do we want to use unicode?
- In which cases do we want definitions to be (fully) typed?
- In which cases do we want accessor functions for all data and properties?

No lines should have trailing whitespace and all files should have a trailing newline.

## CHAPTER 2

### Week 08

#### 1. Univalent Categories

A univalent category is a category in which the univalence axiom holds. I.e., a category  $\mathcal{C}$  in which, for all  $A, B \in \mathcal{C}_0$ , the canonical map  $(A =_{\mathcal{C}} B) \rightarrow (A \cong B)$  is an equivalence.

#### 2. Categories

An  **$n$ -category** is a category with 0-cells (objects), 1-cells (morphisms), 2-cells (morphisms between morphisms), up to  $n$ -cells and various compositions:  $A \rightarrow B \rightarrow C$ .  $A \xrightarrow{f,g,h} B$ ,  $f \Rightarrow g \Rightarrow h$ .  $A \xrightarrow{f,g} B \xrightarrow{f',g'} C$ ,  $\alpha : f \Rightarrow g$ , and identities  $\alpha' : f' \Rightarrow g'$  gives  $\alpha' * \alpha : f' \circ f \Rightarrow g' \circ g$ . These all need to work together ‘nicely’. An  $\omega$ -category is the same, but all the way up.

A topological space gives a (weak)  $\omega$ -category. 0-cells are points, 1-cells are paths, 2-cells are homotopies etc. Composition is by glueing. It is a ‘groupoid’, in the sense that all homotopies of dimension  $\geq 1$  are invertible. However, glueing is not associative, so it is a ‘weak’  $\omega$ -category.

A category with only one object  $\star$  is equivalent to a monoid (with elements being the set  $\mathcal{C}(\star, \star)$ ). A 2-category with only one 0-cell is the same thing as a monoidal category (objects: the 1-cells. Morphisms: the 2-cells). A monoidal category with just one object gives 2 monoid structures on its set of morphisms. These are the same, and commutative.

A **monoid** is a set with a multiplication and a unit. A **monad** on a category  $\mathcal{C}$  is a functor  $T : \mathcal{C} \rightarrow \mathcal{C}$ , together with natural transformations  $\mu : T \circ T \rightarrow T$  (satisfying associativity) and  $\eta : 1_{\mathcal{A}} \rightarrow T$  (acting as a two-sided unit).

A **presheaf** on a category  $\mathcal{A}$  is a functor  $\mathcal{A}^{opp} \rightarrow \mathbf{Set}$ .

Given a category  $\mathcal{E}$  and an object  $E \in \mathcal{E}_0$ , the **slice category**  $\mathcal{E}/E$  with objects being the maps  $D \xrightarrow{p} E$  and morphisms being commutative triangles.

A **multicategory**, not necessarily the same as an  $n$ -category, is a category in which arrows go from multiple objects to one, instead of from one object to one. I.e. it is a category with a class  $C_0$  of objects, for all  $n$ , and all  $a, a_1, \dots, a_n \in C_0$ , a class  $C(a_1, \dots, a_n; a)$  of ‘morphisms’, and a composition

$$C(a_1, \dots, a_n; a) \times C(a_1, 1, \dots, a_{1,k_1}; a_1) \times \dots \times C(a_{n,1}, \dots, a_{n,k_n}; a_n) \rightarrow C(a_1, 1, \dots, a_{n,k_n}; a),$$

written  $(\theta, \theta_1, \dots, \theta_n) \mapsto \theta(\theta_1, \dots, \theta_n)$  and for each  $a \in C_0$  an identity  $1_a \in C(a; a)$ . It must satisfy associativity

$$\theta \circ (\theta_1 \circ (\theta_{1,1}, \dots, \theta_{1,k_1}), \dots, \theta_n \circ (\theta_{n,1}, \dots, \theta_{n,k_n})) = (\theta \circ (\theta_1, \dots, \theta_n)) \circ (\theta_{1,1}, \dots, \theta_{n,k_n})$$

and identity

$$\theta \circ (1_{a_1}, \dots, 1_{a_n}) = \theta = 1_a \circ \theta.$$

A **map of multicategories** is a function  $f_0 : C_0 \rightarrow C'_0$  and maps  $C(a_1, \dots, a_n; a) \rightarrow C(f_0(a_1), \dots, f_0(a_n); f_0(a))$ , preserving composition and identities.

For  $C$  a multicategory, a  **$C$ -algebra** is a map from  $C$  into the multicategory **Set** (with objects **Set**<sub>0</sub> and maps **Set** $(a_1, \dots, a_n; a) = \mathbf{Set}(a_1 \times \dots \times a_n; a)$ ). I.e., for each  $a \in C_0$ , a set  $X(a)$ , and for each map  $\theta : a_1, \dots, a_n \rightarrow a$ , a function  $X(\theta) : X(a_1) \times \dots \times X(a_n) \rightarrow X(a)$ . An example is, for a multicategory  $C$ , to take  $X(a) = C(; a)$  (maps from the empty sequence into  $a$ ).

Of course, there is a concept of **free multicategory**: Given a set  $X$ , and for all  $n \in \mathbb{N}$ , and  $x_1, \dots, x_n \in X$ , a set  $X(x_1, \dots, x_n; x)$ , we get a multicategory  $X'$  with  $X'_0 = X_0$ , and  $X'(x_1, \dots, x_n; x)$  given by formal compositions of elements of the  $X(y_1, \dots, y_m; y)$ .

A **bicategory** consists of a class  $\mathcal{B}_0$  of 0-cells, or objects; For each  $A, B \in \mathcal{B}_0$ , a category  $\mathcal{B}(A, B)$  of 1-cells (objects) and 2-cells (morphisms); for each  $A, B, C \in \mathcal{B}_0$ , a functor  $\mathcal{B}(B, C) \times \mathcal{B}(A, B) \rightarrow \mathcal{B}(A, C)$  written  $(g, f) \mapsto g \circ f$  on 1-cells and  $(\delta, \gamma) \mapsto \delta * \gamma$  on 2-cells; For each  $A \in \mathcal{B}_0$  an object  $1_A \in \mathcal{B}(A, A)$ ; isomorphisms representing associativity and identity axioms (e.g.  $f \circ 1_A \cong f \in \mathcal{B}(A, B)$ ), natural in their arguments, satisfying pentagon and triangle axioms.

The collection of categories **Cat** forms a bicategory. In analogy, we define a monad in a bicategory to be an object  $A$ , together with a 1-cell  $t : A \rightarrow A$  and 2-cells  $\mu : t \circ t \rightarrow t$  and  $\eta : 1_A \rightarrow t$  satisfying a couple of commutativity axioms (those of 1.1.3 in [Lei03]).

### 3. Operads

**3.1. Definitions.** An **operad** is a multicategory with only one object. More explicitly, an operad has a set  $P(k)$  for every  $k \in \mathbb{N}$ , whose elements can be thought of as  $k$ -ary operations. It also has, for all  $n, k_1, \dots, k_n \in \mathbb{N}$ , a *composition* function

$$P(n) \times P(k_1) \times \dots \times P(k_n) \rightarrow P(k_1 + \dots + k_n)$$

and an element  $1 = 1_P \in P(1)$  called the **identity**, satisfying

$$\theta \circ (1, 1, \dots, 1) = \theta = \theta \circ 1$$

for all  $\theta$ , and

$$\theta \circ (\theta_1 \circ (\theta_{1,1}, \dots, \theta_{1,k_1}), \dots, \theta_n \circ (\theta_{n,1}, \dots, \theta_{n,k_n})) = (\theta \circ (\theta_1, \dots, \theta_n)) \circ (\theta_{1,1}, \dots, \theta_{n,k_n})$$

for all  $\theta \in P(n)$ ,  $\theta_1 \in P(k_1)$ ,  $\dots$ ,  $\theta_n \in P(k_n)$  and all  $\theta_{1,1} \dots \theta_{n,k_n}$ .

A **morphism of operads** is a family

$$f_n : (P(n) \rightarrow Q(n))_{n \in \mathbb{N}}$$

of functions, preserving composition and identities.

A  **$P$ -algebra** for  $P$  an operad, is a set  $X$  and, for each  $n$ , and  $\theta \in P(n)$ , a function  $\bar{\theta} : X^n \rightarrow X$ , satisfying the evident axioms (identity is the identity function, the function of a composition is the composition of the functions?).

**3.2. Examples.** For any vector space  $V$ , there is an operad with  $P(k) = V^{\otimes k} \rightarrow V$ .

The terminal operad **1** has  $P(n) = \{\star_1\}$  for all  $n$ . An algebra for **1** is a set  $X$  together with a function  $X^n \rightarrow X$ , denoted as  $(x_1, \dots, x_n) \mapsto (x_1 \cdots x_n)$ , satisfying

$$((x_{1,1} \cdots x_{1,k_1}) \cdots (x_{n,1} \cdots x_{n,k_n})) = (x_{1,1} \cdots x_{n,k_n})$$

and

$$x = (x).$$

The category of 1-algebras is the category of monoids.

There exist various sub-operads of 1. For example, the smallest one has  $P(1) = \{\star\}$  and  $P(n) = \emptyset$  for  $n \neq 1$ .

Or the operad with  $P(0) = \emptyset$  and  $P(n) = \{\star_n\}$  for  $n > 0$ , which has semigroups as its algebras (sets with associative binary operations).

The suboperad with  $P(n) = \{\star_n\}$  exactly when  $n \leq 1$  has as its algebras the pointed sets.

The **operad of curves** has  $P(n) = \{\text{smooth maps } \mathbb{R} \rightarrow \mathbb{R}^n\}$ .

Given a monad on **Set**, we get a natural operad structure  $T(n)_{n \in \mathbb{N}}$ , with  $T(n)$  the set of words in  $n$  variables and composition given by ‘substitution’.

Given a monoid  $M$  (a category with one object), there is an operad given by  $P(n) = M^n$  and composition

$$(\alpha_1, \dots, \alpha_n) \circ ((\alpha_{1,1}, \dots, \alpha_{1,k_1}), \dots, (\alpha_{n,1}, \dots, \alpha_{n,k_n})).$$

The **Little 2-disks** operad  $D$  has

$$D(n) = \{\text{set of non-overlapping disks contained within the unit disk}\},$$

with composition being geometric “substitution”. I.e.: scale and move a unit disk and its contained disks to match one of the smaller disks, and replace the smaller disk with the transformed contents of our original unit disk. See also: this image that explains a lot

Given sets  $X(n)$  for all  $n \in \mathbb{N}$ , the **free operad**  $X'$  on these is defined exactly by  $X(n) \subseteq X'(n)$ ,  $1 \in X'(1)$  and for all  $m, n_1, \dots, n_m \in \mathbb{N}$  and  $f \in X(m)$  and  $f_i \in X'(n_i)$ , we have  $f \circ (f_1, \dots, f_m) \in X'(n_1 + \dots + n_m)$ .

## 4. T-operads

**4.1. Definitions.** A category is **cartesian** if it has all pullbacks. A functor is cartesian if it preserves pullbacks. A natural transformation  $\alpha : S \rightarrow T$  is cartesian if for all  $f : A \rightarrow B$ , the naturality diagram

$$\begin{array}{ccc} SA & \xrightarrow{Sf} & SB \\ \downarrow \alpha_A & & \downarrow \alpha_B \\ TA & \xrightarrow{Tf} & TB \end{array}$$

is a pullback. A monad  $(T, \mu, \eta)$  on a category  $\mathcal{E}$  is cartesian if the category  $\mathcal{E}$ , the functor  $T$  and the natural transformations  $\mu$  and  $\eta$  are cartesian.

We can represent (the morphism structure of) an ordinary category using diagrams  $C_0 \xleftarrow{\text{domain}} C_1 \xrightarrow{\text{codomain}} C_0$ ,  $C_1 \times_{C_0} C_1 \xrightarrow{\text{composition}} C_1$  and  $C_0 \xrightarrow{\text{id}} C_1$  together with some axioms. For a multicategory, we need to slightly modify this, using a functor  $T : \mathbf{Set} \rightarrow \mathbf{Set}$ ,  $A \mapsto \bigsqcup A^n$ , to  $TC_0 \xleftarrow{d} C_1 \xrightarrow{c} C_0$  and  $C_1 \times_{TC_0} TC_1 \xrightarrow{\circ} C_1$ .

Given a cartesian monad  $(T, \mu, \eta)$  on a category  $\mathcal{E}$ , we can define a bicategory  $\mathcal{E}_{(T)}$ , with the class of 0-cells being  $\mathcal{E}_0$ , the 1-cells  $E \rightarrow E'$  being diagrams  $TE \xleftarrow{d} M \xrightarrow{c} E'$ , 2-cells  $(M, d, c) \rightarrow (N, q, p)$  are maps  $M \rightarrow N$  such that the diagram with  $E, E', M, N$  commutes. The composite of 1-cells  $TE \xleftarrow{d} M \xrightarrow{c} E'$  and  $TE \xleftarrow{d'}$

$M' \xrightarrow{c'} E''$  is given by

$$TE \xleftarrow{\mu_E} T^2E \xleftarrow{Td} TM \leftarrow TM \times_{TE'} M' \rightarrow M' \xrightarrow{c'} E''$$

in which the coproduct in the middle is formed using  $Tc$  and  $d$ . We can define a  $T$ -multicategory to be a monad on  $\mathcal{E}_{(T)}$ . Equivalently, we can define it as an object  $C_0 \in \mathcal{E}$ , together with a diagram  $t : TC_0 \xleftarrow{d} C_1 \xrightarrow{c} C_0$  and maps  $C_1 \circ C_1 := TC_1 \times_{TC_0} C_1 \xrightarrow{\circ} C_1$  and  $C_0 \xrightarrow{id} C_1$  satisfying associativity and identity axioms.

A  $T$ -operad is a  $T$ -multicategory such that  $C_0$  is the terminal object of  $\mathcal{E}$ . Equivalently, it is an object over  $T1$ , (so we have a morphism  $P \rightarrow T1$ ), together with maps  $P \times_{T1} TP \rightarrow P$  and  $1 \xrightarrow{id} P$ , both over  $T1$ , satisfying associativity and identity axioms.

**4.2. Examples.** For  $T$  the identity monad on **Set**, a  $T$ -operad is exactly a monoid (or an operad with only unary functions) (since there is always a unique map to  $\{1\}$ ).

If  $\mathcal{E}$  is **Set**, the terminal object  $1$  will always be  $\{1\}$ .

For the free monoid monad  $TA = \bigsqcup A^n$ , the  $T$ -operads are precisely the operads that we defined before.

For the monad  $TA = 1 + A$ , we can view  $TA$  as a subset of the free monoid on  $A$ , and this gives an operad with 0-ary and 1-ary functions. The 1-ary arrows form a monoid, and the 0-ary arrows are a set, with an action of the monoid.

## 5. Cartesian Operads

**5.1. Theory.** Using Towards a doctrine of operads.

NLab uses notation: **Fin** for what we would call a standard skeleton of finite sets (i.e. the category of finite sets  $\{0, \dots, n-1\}$  and maps between them).  $A^B$  denotes all morphisms/functors  $B \rightarrow A$ . I.e., the class of functors  $\mathbf{Fin} \rightarrow \mathbf{Set}$  is denoted  $\mathbf{Set}^{\mathbf{Fin}}$ .

Take  $I = \mathbf{Fin}(1, -) : \mathbf{Set}^{\mathbf{Fin}} = \mathbf{Fin} \rightarrow \mathbf{Set}$ .

Let  $[\mathbf{Set}^{\mathbf{Fin}}, \mathbf{Set}^{\mathbf{Fin}}]$  be the category of finite-product-preserving, cocontinuous endofunctors on  $\mathbf{Set}^{\mathbf{Fin}}$ . The map  $Ev_I : [\mathbf{Set}^{\mathbf{Fin}}, \mathbf{Set}^{\mathbf{Fin}}] \rightarrow \mathbf{Set}^{\mathbf{Fin}}$ , given by  $F \mapsto F(I)$  is an equivalence.  $[\mathbf{Set}^{\mathbf{Fin}}, \mathbf{Set}^{\mathbf{Fin}}]$  has a monoidal product  $\odot$  given by endofunctor composition, and we can transfer this to  $\mathbf{Set}^{\mathbf{Fin}}$ .

Concretely, we have  $F \odot G = \int^{n \in \mathbf{Fin}} F(n)G^n$ .

**5.2. Cartesian operads.** A cartesian operad is a monoid in this monoidal category  $(\mathbf{Set}^{\mathbf{Fin}}, \odot, I)$ . I.e., it is a triple  $(M, \mu, \eta)$ , with  $M \in \mathbf{Set}^{\mathbf{Fin}}$ ,  $\mu : M \odot M \rightarrow M$  and  $\eta : I \rightarrow M$ .

More concretely, this is a functor  $M : \mathbf{Fin} \rightarrow \mathbf{Set}$ , together with maps

$$m_{n,k} : M(n) \times M(k)^n \rightarrow M(k),$$

natural in  $k$  and dinatural in  $n$ , and an element  $e \in M(1)$ .

Dinaturality in  $n$  means the following. Fix  $k \in \mathbf{Fin}$ . We have the functors  $\mathbf{Fin}^{\text{op}} \times \mathbf{Fin} \rightarrow \mathbf{Set}$  given by

$$F : (n, n') \mapsto M(n') \times M(k)^n \quad \text{and} \quad G : (n, n') \mapsto M(k).$$

For all  $n \in \mathbf{Fin}_0$ , we have a morphism

$$\bullet : F(n, n) = M(n) \times M(k)^n \rightarrow M(k) = G(n, n).$$



Naturality means that for all  $a : n \rightarrow n'$ ,

$$G(n, a) \circ \bullet \circ F(a, n) = F(a, n') \circ \bullet \circ G(n', a).$$

i.e., for all  $f \in M(n)$ ,  $g_1, \dots, g_{n'} \in M(k)$ ,

$$f \bullet (g_{a(1)}, \dots, g_{a(n)}) = M(a)(f) \bullet (g_1, \dots, g_{n'}).$$

Now, if we have, in **Fin**, a decomposition  $k = k_1 + \dots + k_n$ , and we have inclusion maps  $i_j : k_j \hookrightarrow k$ , then we have

$$M(n) \times M(k_1) \times \dots \times M(k_n) \xrightarrow{1 \times M(i_1) \times \dots \times M(i_n)} M(n) \times M(k)^n \xrightarrow{m_{n,k}} M(k),$$

which gives an operad structure.

**5.3. Clones.** In other parts of mathematics, a cartesian operad is called a **clone**. An abstract clone consists of sets  $M(n)$  for all  $n \in \mathbb{N}$ , for all  $n, k \in \mathbb{N}$  a function  $\bullet : M(n) \times M(k)^n \rightarrow M(k)$  and for each  $1 \leq i \leq n \in \mathbb{N}$ , an element  $\pi_{i,n} \in M(n)$  such that for  $f \in M(i)$ ,  $g_1, \dots, g_i \in M(j)$  and  $h_1, \dots, h_j \in M(k)$ ,

$$f \bullet (g_1 \bullet (h_1, \dots, h_j), \dots, g_i \bullet (h_1, \dots, h_j)) = (f \bullet (g_1, \dots, g_i)) \bullet (h_1, \dots, h_j),$$

for  $f_1, \dots, f_n \in M(k)$ ,

$$\pi_{i,n} \bullet (f_1, \dots, f_n) = f_i,$$

and for  $f \in M(n)$ ,

$$f \bullet (\pi_{1,n}, \dots, \pi_{n,n}) = f.$$

(It is claimed that this automatically gives naturality)

Naturality means for all  $a \in \mathbf{Fin}(n, n')$ , for all  $f \in M(n)$ ,  $g_1, \dots, g_{n'} \in M(k)$ ,

$$f \bullet (g_{a(1)}, \dots, g_{a(n)}) = (f \bullet (\pi_{a(1),n'}, \dots, \pi_{a(n),n'})) \bullet (g_1, \dots, g_{n'}).$$

However, by associativity and since  $\pi_{i,n} \bullet (f_1, \dots, f_n)$ , we have

$$\begin{aligned} & (f \bullet (\pi_{a(1),n'}, \dots, \pi_{a(n),n'})) \bullet (g_1, \dots, g_{n'}) \\ &= (f \bullet (\pi_{a(1),n'} \bullet (g_1, \dots, g_{n'}), \dots, \pi_{a(n),n'} \bullet (g_1, \dots, g_{n'}))) \\ &= f \bullet (g_{a(1)}, \dots, g_{a(n)}). \end{aligned}$$

Naturality in  $k$  is as follows. Fix  $n \in \mathbf{Fin}_0$ . We have functors  $F, G : \mathbf{Fin} \rightarrow \mathbf{Set}$ , given by

$$F : k \mapsto M(n) \times M(k)^n \quad \text{and} \quad G : k \mapsto M(k).$$

For  $a : k \rightarrow k'$ , we must have

$$G(a) \circ \bullet = \bullet \circ F(a).$$

That is, for all  $f \in M(n)$ ,  $g_1, \dots, g_n \in M(k)$ ,

$$M(a)(f \bullet (g_1, \dots, g_n)) = f \bullet (M(a)(g_1), \dots, M(a)(g_n)).$$

Now,  $M(a)$  is given by

$$M(a)(f) = f \bullet (\pi_{a(1),k'}, \dots, \pi_{a(k),k'}).$$

Therefore, we have

$$\begin{aligned} & M(a)(f \bullet (g_1, \dots, g_n)) \\ &= (f \bullet (g_1, \dots, g_n)) \bullet (\pi_{a(1),k'}, \dots, \pi_{a(k),k'}) \\ &= f \bullet (g_1 \bullet (\pi_{a(1),k'}, \dots, \pi_{a(k),k'}), \dots, g_n \bullet (\pi_{a(1),k'}, \dots, \pi_{a(k),k'})) \\ &= f \bullet (M(a)(g_1), \dots, M(a)(g_n)). \end{aligned}$$

So the associativity and projection axioms ensure naturality.

## CHAPTER 3

### Week 09

#### 1. Adjunctions

For three sets  $X, Y, Z$ , we have a bijection

$$\mathbf{Set}(X \times Y, Z) \cong \mathbf{Set}(X, Y \rightarrow Z)$$

that is natural in  $X$  and  $Z$ . This is an example of a general notion:

For functors  $F : \mathcal{C} \rightarrow \mathcal{D}$  and  $G : \mathcal{D} \rightarrow \mathcal{C}$ ,  $F$  is a **left adjoint** for  $G$  if there is an isomorphism  $\mathcal{C}(FX, Y) \cong \mathcal{D}(X, GY)$  that is dinatural in  $X$  and  $Y$ .

**1.1. Examples.** A special case is for  $G : \mathcal{C} \rightarrow \mathbf{Set}$  the forgetful functor: if we have a left adjoint  $F : \mathbf{Set} \rightarrow \mathcal{C}$  to  $G$ , we call  $F(X)$  the **free**  $\mathcal{C}$  of  $X$  (for example: free group, free monoid, free category, etc.). That means that  $\mathcal{C}(F(X), Y) \cong \mathbf{Set}(X, G(Y))$ , or in other words:

Adjunctions can be composed: If  $F$  and  $G$  both forget part of the structure of an object, then we can construct the free object ‘piecewise’ using the composition of the left adjoints.

Let  $G : \mathbf{Group} \rightarrow \mathbf{Set}$  be the forgetful functor and  $F : \mathbf{Set} \rightarrow \mathbf{Group}$  be the free group functor. For any set  $S$ , we have an inclusion  $i : S \hookrightarrow FS$ . Now, given any group  $Y$  and any morphism of sets from  $S$  to  $Y$  (formally:  $f \in \mathbf{Set}(S, GY)$ ). Then the fact that  $\mathbf{Set}(FS, Y) \cong \mathbf{Set}(S, GY)$  is expressed in the fact that  $f$  factors uniquely as  $g \circ i$ .

$$\begin{array}{ccc} S & \xrightarrow{i} & F(S) \\ & \searrow f & \downarrow \exists! g \\ & & Y \end{array}$$

In topology, the forgetful functor from topological spaces to sets has a left adjoint that turns a set  $S$  into a space  $F(S)$  with the discrete topology, since it has the smallest topology on  $S$  that still can factor all morphisms  $S \rightarrow [0, 1]$  for example.

In algebra, the inclusion functor from the category of monoids (sets with an associative operation and a unit) to the category of semigroups (sets with an associative operation) has a left adjoint that sends the semigroup  $S$  to the monoid  $S \sqcup \{\star\}$  and gives it operations such that  $\star$  is the unit.

In algebra, the inclusion functor from the category of rings to the category of rngs (rings but without an identity) has a left adjoint that sends the rng  $R$  to  $R \times \mathbb{Z}$  and defines  $(r, x)(s, y) = (rs + xs + ry, xy)$ .

For  $\mathbf{Dom_m}$  the category of integral domains with injective morphisms. Then the forgetful functor  $\mathbf{Fields} \rightarrow \mathbf{Dom_m}$  has a left adjoint that sends a domain to its field of fractions.

For  $\rho : R \rightarrow S$  a morphism of rings, we can view every  $S$ -module as an  $R$ -module: we have a functor  $S\text{-}\mathbf{Mod} \rightarrow R\text{-}\mathbf{Mod}$ . This has a left adjoint,  $M \mapsto M \otimes_R S$ .

## 2. Operads and cartesian operads

**2.1. Recap operads and cartesian operads.** In this section, we will represent a cartesian operad  $C$  by a clone:

- For all  $n \in \mathbb{N}$ , a set  $C(n)$ ;
- For all  $1 \leq i \leq n$  projections  $\pi_{n,i} \in P(n)$ ;
- For all  $m, n$ , a function  $\rho_{m,n} : C(m) \times C(n)^m \rightarrow C(n)$ .

such that

- For  $f \in C(l)$ ,  $g_1, \dots, g_l \in C(m)$  and  $h_1, \dots, h_m \in C(n)$ ,

$$\rho(f, (\rho(g_1, (h_1, \dots, h_m)), \dots, \rho(g_l, (h_1, \dots, h_m)))) = \rho(\rho(f, (g_1, \dots, g_l)), (h_1, \dots, h_l)).$$

- for  $f_1, \dots, f_n \in M(n)$  and  $1 \leq i \leq n$ ,  $\rho(\pi_{n,i}, (f_1, \dots, f_n)) = f_i$ ;
- For  $f \in M(n)$ ,  $\rho(f, (\pi_{1,n}, \dots, \pi_{n,n})) = f$ .

and a morphism of cartesian operads  $\varphi : C \rightarrow C'$  by componentwise functions  $\varphi_n : C(n) \rightarrow C'(n)$  such that

- For all  $1 \leq i \leq n$ ,  $\varphi_n(\pi_{n,i}) = \pi'_{n,i}$ ;
- For all  $m, n$ ,  $f \in C(m)$ ,  $g_1, \dots, g_m \in C(n)$

$$\rho(\varphi(f), (\varphi(g_1), \dots, \varphi(g_m))) = \varphi(\rho(f, (g_1, \dots, g_m))).$$

We will represent an operad  $P$  by

- For all  $n \in \mathbb{N}$ , a set  $P(n)$ ;
- An element  $e \in P(1)$ ;
- For all  $m, n_1, \dots, n_m \in \mathbb{N}$ , a function

$$\gamma_{m,n_1,\dots,n_m} : C(m) \times C(n_1) \times \dots \times C(n_m) \rightarrow C(n_1 + \dots + n_m)$$

(we will usually just call this  $\gamma$ ).

such that

- For all  $f \in P(n)$ ,

$$\gamma(f, (1, \dots, 1)) = f = \gamma(1, f);$$

- For all  $f \in P(l)$ ,  $g_1 \in P(m_1), \dots, g_l \in P(m_l)$  and all  $h_{1,1} \in P(n_{1,1}), \dots, h_{l,m_l} \in P(n_{l,m_l})$ ,

$$f \circ (g_1 \circ (h_{1,1}, \dots, h_{1,m_1}), \dots, g_l \circ (h_{l,1}, \dots, h_{l,m_l})) = (f \circ (g_1, \dots, g_l)) \circ (h_{1,1}, \dots, h_{l,m_l}).$$

and a morphism of operads  $\psi : P \rightarrow P'$  by componentwise functions  $\psi_n : P(n) \rightarrow P'(n)$  such that

- $\psi_1(e) = e'$ ;
- For all  $f \in P(m)$ ,  $g_1 \in P(n_1), \dots, g_m \in P(n_m)$ ,

$$\gamma(\psi(f), (\psi(g_1), \dots, \psi(g_m))) = \psi(\gamma(f, (g_1, \dots, g_m))).$$

**2.2. Free operad?** Now, for all  $1 \leq i \leq m$ , we have an injection  $j_i : C(n_i) \rightarrow C(n_1 + \dots + n_m)$ . Intuitively, it maps terms in a context with  $n_i$  variables to terms in a context with  $n_1 + \dots + n_m$  variables, by mapping variable  $1 \leq k \leq n_1$  to variable  $n_1 + \dots + n_{i-1} + k$ . This gives a morphism

$$\gamma : C(m) \times C(n_1) \times \dots \times C(n_m) \xrightarrow{id \times j_1 \times \dots \times j_m} C(m) \times C(n_1 + \dots + n_m)^m \xrightarrow{\rho} C(n_1 + \dots + n_m),$$

which gives an operad structure on the same sets. This gives a functor from cartesian operads to operads.

Then, the question arises whether this functor has a left adjoint.

**Still open**

### 3. Cartesian multicategory

Yet another way to think of a cartesian operad is as a one-object cartesian multicategory. A **cartesian multicategory** is a multicategory with an  $S_n$ -action on the hom-sets (in other words: we can permute the arguments of the morphisms) and duplication/diagonal/contraction operations

$$\text{Hom}(c_1, \dots, c_k, c_k, \dots, c_n; c) \rightarrow \text{Hom}(c_1, \dots, c_k, \dots, c_n; c)$$

and deletion/projection/weakening operations

$$\text{Hom}(c_1, \dots, c_k, \dots, c_n; c) \rightarrow \text{Hom}(c_1, \dots, c_{k-1}, c_{k+1}, \dots, c_n; c)$$

### 4. The paper

A summary of the results in the paper:

- (1) A definition of the category of algebraic theories (clones).
- (2) A definition of the category of  $T$ -algebras (for  $T$  an algebraic theory), the pullback of an algebra, the free algebra, and some properties of these.
- (3) A definition of a presheaf (like an algebra, but with a right action instead of a left one) and some properties of this.
- (4) A sidenote about the more classical approach to algebra.
- (5) A definition for a  $\lambda$ -theory and some properties of its constituents.
- (6) A definition for an interpretation of the  $\lambda$ -calculus in a  $\lambda$ -theory, and some properties.
- (7) The notion that the algebraic theory  $\Lambda$  of all terms of the  $\lambda$ -calculus is the initial  $\lambda$ -theory.
- (8) The notion that we can add constants to a theory to make sure that it ‘has enough points’.
- (9) A (new) proof that any  $\lambda$ -theory is isomorphic to the endomorphism  $\lambda$ -theory of some object.
- (10) A section that concludes an equivalence between the presheaf categories of a Lawvere theory, a  $\lambda$ -theory and the category of retracts.
- (11) The definition of  $\Lambda$ -algebra and a couple of its properties. In particular, a functor from  $\lambda$ -theories to  $\Lambda$ -algebras.
- (12) An equivalence between  $\Lambda$ -algebras and their presheaves.
- (13) A characterisation of the function space  $U^U$  for  $U \in PA$  the universal.
- (14) An equivalence of categories between the category of  $\Lambda$ -algebras and  $\lambda$ -theories.
- (15) The Fundamental Theorem of the  $\lambda$ -calculus: there is an adjoint equivalence between  $\lambda$ -theories and  $\Lambda$ -algebras.

- (16) A remark that this is much harder without the category theoretic framework.

## CHAPTER 4

### Week 11

#### 1. A Formalization of Operads in Coq

A paper was uploaded to the arXiv (see [FTBF23]). Unfortunately, this paper did not contain any code, which makes it harder to compare their technique to mine. The paper is about the formalization of a symmetric multicategory (kind of). They do not require  $(\tau\sigma)f = \tau(\sigma f)$  for  $\sigma, \tau \in S_n$  and  $f \in \mathcal{C}(a_1, \dots, a_n; a)$

The operad data presented in this paper is a bit different (but equivalent) from the data that I have seen so far. Notably: the composition is on one element at a time:

$$\mathcal{C}(a_1, \dots, a_n; a) \times \mathcal{C}(b_1, \dots, b_m; a_i) \rightarrow \mathcal{C}(a_1, \dots, a_{i-1}, b_1, \dots, b_m, a_{i+1}, \dots, a_n; a),$$

in which  $a_1, \dots, a_{i-1}, b_1, \dots, b_m, a_{i+1}, \dots, a_n$  is denoted  $\underline{a} \bullet_i \underline{b}$ .

I expect the axioms to become a bit more complicated this way. For example, now there are two associativity axioms, and even to state them we need a proof that  $(\underline{c} \bullet_i \underline{a}) \bullet_{l-1+j} \underline{b} = (\underline{c} \bullet_j \underline{b}) \bullet_i \underline{a}$ .

Also, their signature of  $Hom$  is **Type**  $\rightarrow$  **List Type**  $\rightarrow$  **Type**. The advantage of using **List Type** instead of  $\sum_{n \in \mathbb{N}} (\mathbf{stn} \ n \rightarrow \mathbf{Type})$  is that it sounds simpler. The disadvantage is that, for example, to say something about  $c_i$ , one needs a function to fetch the  $i$ th element of  $\underline{c}$  and a proof that  $i$  is less than the length of  $\underline{c}$ .

To get elements in the right space, they add a lot of typecasting functions. For example, the function  $\mathcal{C}_{assoc} : \mathcal{O}((\underline{c} \bullet_i \underline{a}) \bullet_{l-1+j} \underline{b}; d) \rightarrow \mathcal{O}((\underline{c} \bullet_j \underline{b}) \bullet_i \underline{a}; d)$  I believe we call such a function a ‘transport function’. The paper claims that if  $A = B$ , a typecast function will be the identity. The way that it is currently stated, it is not compatible with univalence, I believe.

I feel like the paper lacks a section explaining the choices that were made and evaluating their results.





## Bibliography

- [FTBF23] Zachary Flores, Angelo Taranto, Eric Bond, and Yakir Forman. A formalization of operads in coq, 2023.
- [Lei03] Tom Leinster. Higher operads, higher categories, 2003.