Semantics for the λ-calculus

Contents

CHAPTER 1

Category Theoretic Preliminaries

I will assume a familiarity with the category-theoretical concepts presented in [**AW23**]. These include categories, functors, isomorphisms, natural transformations, adjunctions, equivalences and limits.

1. Notation

For an object c in a category C, I will write $c : C$.

For a morphism f between objects c and c' in a category C, I will write $f : C(c, c')$ or $f : c \to c'$.

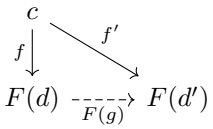
For composition of morphisms $f : C(c, d)$ and $g : C(d, e)$, I will write $f \cdot g$.

For composition of functors $F : A \to B$ and $G : B \to C$, I will write $F \bullet G$.

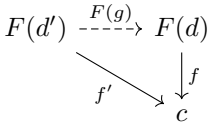
2. Universal Arrows

One concept in category theory that can be used to describe a lot of limits and adjunctions is that of a universal arrow (see for example [**ML98**], Part III)

Definition 1. A *universal arrow* from an object $c : C$ to a functor $F : D \to C$ consists of an object $d : D$ and a morphism $f : D(c, F(d))$ such that for every similar pair (d', f'), f' factors uniquely as $f \cdot F(g)$ for some $g : C(d, d')$:

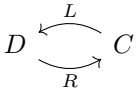


There is also the dual concept of a universal arrow (d, f) from a functor F to an object $c : C$. Its universal property yields the following diagram:



3. Adjunctions

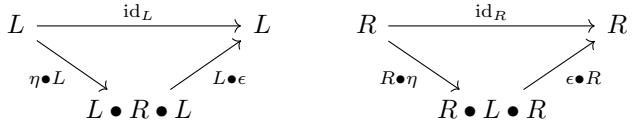
Recall that an *adjunction* $L \dashv R$ is a pair of functors



with natural transformations (the unit and counit)

$$\eta : \mathrm{id}_C \Rightarrow L \bullet R \quad \text{and} \quad \epsilon : R \bullet L \Rightarrow \mathrm{id}_D$$

such that the diagrams



commute. Here the natural transformation $\eta \bullet L : L \bullet R \bullet L$ is the natural transformation η whiskered on the right by L, and the other whiskered transformations are similar.

An alternative characterization (see [**ML98**], chapter IV.1, Theorem 2) of an adjunction $L \dashv R$ is as a natural bijection

$$\varphi : D(L(c), d) \xrightarrow{\sim} C(c, R(d)).$$

Naturality means that for all $f : C(c', c)$, $g : D(d, d')$ and $h : D(L(c), d)$,

$$\varphi(L(f) \cdot h \cdot g) = f \cdot \varphi(h) \cdot R(g).$$

Lastly, one can construct an adjunction using universal arrows. This lends itself particularly well for a formalization, where it is often preferable to have as little 'demonstranda' as possible:

Lemma 1. *One can construct an adjunction (F, G, η, ϵ) as above from only the functor $L : C \to D$ and, for each $c : C$, a universal arrow $(R(c), \epsilon_c)$ from L to c.*

Proof. See [**ML98**], Chapter IV.1, Theorem 2 (iv). □

3.1. Adjoint equivalences. An (adjoint) equivalence of categories has multiple definitions. The one we will use here is the following:

Definition 2. An *adjoint equivalence* between categories C and D is a pair of adjoint functors $L \dashv R$ like above such that the unit $\eta : \mathrm{id}_C \Rightarrow L \bullet R$ and counit $\epsilon : R \bullet L \Rightarrow \mathrm{id}_D$ are isomorphisms of functors.

3.2. Exponential objects. Note that in the category of sets, for all $X, Y :$ **Set**, we have a set $(X \to Y)$. Also, for all X, Y, Z, there is a (natural) bijection

$$(X \times Y \to Z) \cong (X \to (Y \to Z))$$

which we can also write as

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

In other words, we have functors $X \mapsto X \times Y$ and $Z \mapsto (Y \to Z)$, and these two form an adjunction. The following generalizes this

Definition 3. A category C has *exponential objects* (or *exponentials*) if for all $c : C$, the functor $c' \mapsto c' \times c$ has a right adjoint, which we denote $d \mapsto d^c$.

Remark 1. It is actually very well possible that a category does not have all exponentials, but it has some objects $c, d, d^c : C$ with a natural bijection

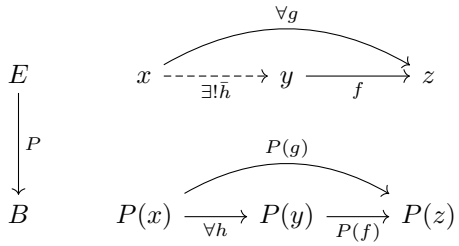
$$C(d' \times c, d) \cong C(d', d^c).$$

Then d^c is still called an exponential object.

4. Fibrations

Let $P : E \to B$ be a functor. In this case, we will view this as the category E 'lying over' the category B, with for every point $b : B$, a slice $E_b = P^{-1}(B)$ lying 'above' b.

Definition 4. A morphism $f : E(y, z)$ is called *cartesian* if for all $g : E(x, z)$ and $h : B(P(x), P(y))$ with $h \cdot P(f) = P(g)$, there exists $\bar{h} : E(x, y)$ such that $P(\bar{h}) = h$ and $\bar{h} \cdot f = g$, like illustrated in the following diagram from [**nLa24**]



DEFINITION 5. P is a *fibration* if for all $y : E$ and morphisms $f : B(x, P(y))$, there exist an object $\bar{x} : E$ and a cartesian morphism $\bar{f} : E(\bar{x}, y)$ such that $P(\bar{x}) = x$ and $P(\bar{f}) = f$:

$$\begin{array}{ll} E & \bar{x} \overset{\exists \bar{f}}{\dashrightarrow} y \\ \downarrow P & \\ B & x \overset{\forall f}{\longrightarrow} P(y) \end{array}$$

5. (Co)slice categories

Given an object in a category $c : C$, the morphisms to and from c constitute the slice and coslice categories

DEFINITION 6. The *slice category* $C \downarrow c$ is the category with as objects the morphisms to c:

$$(C \downarrow c)_0 = \sum_{c':C} C(c', c).$$

The morphisms from (c', f) to (c'', f') are the morphisms $g : c' \to c''$ making the following diagram commute.

$$\begin{array}{ccc} c' & \xrightarrow{g} & c'' \\ & {\scriptstyle f}\searrow \quad \swarrow{\scriptstyle f'} & \\ & c & \end{array}$$

The *coslice category* $c \downarrow C$ is similar, but with the morphisms *from* c instead of *to* c:

$$(c \downarrow C)_0 = \sum_{c':C} C(c, c').$$

6. Dependent products and sums

The following is based loosely on Section 4.1 of [**Tay86**].

Take a category C. To talk about dependent sums $\sum_{a:A} X_a$ and products $\prod_{a:A} X_a$ in C, we first need some way to construct the family of objects $(X_a)_a$. Of course, we can do this *externally* using a set A, and picking an object $X_a : C$ for every element $a : A$. However, there is also an *internal* representation, as a morphism $X \to A$.

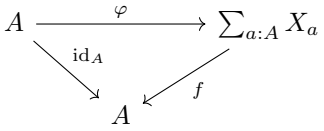
Note that in **Set**, the external and internal ways of indexing are equivalent, because given a family $(X_a)_a$ we can construct a morphism $f : \sum_{a:A} X_a \to A$ and conversely, we can recover the family $(X_a)_a$ as $(f^{-1}(a))_a$.

In **Set**, given a family $(X_a)_a$ over A and a morphism $\alpha : B \to A$, we get a family $X_{\alpha(b)}$ over B. It turns out that in the internal representation this arises as the pullback

$$\begin{array}{ccc} \sum_{b:B} X_{\alpha(b)} & \longrightarrow & \sum_{a:A} X_a \\ \downarrow{\scriptstyle \alpha^* f} & \lrcorner & \downarrow{\scriptstyle f} \\ B & \xrightarrow{\alpha} & A \end{array}$$

This can be turned into a pullback or 'substitution' functor α^*, which Taylor calls $\mathtt{P}\alpha$. We can construct $\alpha^* : (C \downarrow A) \to (C \downarrow B)$ for any category C with pullbacks and any morphism $\alpha : C(B, A)$.

Now, in **Set**, for a family $(X_a)_a$, consider the dependent product $\prod_{a:A} X_a$. Its elements $(x_a)_a$ can be identified with morphisms from the terminal set: $\{\star\} \to \prod_{a:A} X_a$, sending $\star$ to $(x_a)_a$. However, they can also be identified with the morphisms $\varphi : A \to \sum_{a:A} X_a$ that make the following diagram commute, sending a to x_a:



These are morphisms in $(\mathbf{Set} \downarrow A)$ from (A, id_A) to $(\sum_{a:A} X_a, f)$. Note that for the terminal morphism $\alpha : A \to \{\star\}$, we have $\mathrm{id}_A = \alpha^*(\mathrm{id}_{\{\star\}})$. To summarize, we have an equivalence

$$(\mathbf{Set} \downarrow A)(\alpha^*(\mathrm{id}_{\{\star\}}), f) \simeq (\mathbf{Set} \downarrow \{\star\})(\mathrm{id}_{\{\star\}}, \prod_{a:A} X_a).$$

Now, given a family of families $((X_b)_{b:B_a})_{a:A}$, we can wonder whether we can construct the family of dependent products $(\prod_{b:B_a} X_b)_a$. In **Set**, this is definitely possible, and from this, we get an equivalence again

$$(\mathbf{Set} \downarrow (\sum_{a:A} B_a))(\alpha^*(\mathrm{id}_A), f) \simeq (\mathbf{Set} \downarrow A)(\mathrm{id}_A, (\prod_{b:B_a} X_b)_a).$$

These equivalences suggest an adjunction $\alpha^* \dashv \prod_{\dots}$. We can use this to define in general

Definition 7. For a category C and a morphism $\alpha : B \to A$, the *dependent product* along α is, if it exists, the right adjoint to the pullback functor:

$$(C \downarrow A) \underset{\prod_\alpha}{\overset{\alpha^*}{\underset{\longleftarrow}{\overset{\longrightarrow}{\bot}}}} (C \downarrow B)$$

Remark 2. As argued above, we can recover the familiar dependent product $\prod_{a:A} X_a$ of a family $(X_a)_a$ as the dependent product $\prod_\alpha f$ along the terminal morphism $\alpha : A \to I$, with $f : \sum_a X_a \to A$ the internal representation of the family. Here we use the equivalence between $(C \downarrow I)$ and C.

Now we turn our attention to dependent sums. In **Set**, let $(X_a)_a$ and $(B_a)_a$ be two families over A and let $((Y_b)_{b:B_a})_{a:A}$ be a family of families. Let $\alpha : \sum_{a:A} B_a \to A$ be the internal representation of $(B_a)_a$. A family of maps $f_a : (\sum_{b:B_a} Y_b)_a \to X_a$ consists of maps $Y_b \to X_a$ for all $b : B_a$, so these are maps $f_b : Y_b \to X_{\alpha(b)}$. This gives an equivalence

$$(\mathbf{Set} \downarrow A)((\sum_{b:B_a} Y_b)_a, (X_a)_a) \simeq (\mathbf{Set} \downarrow (\sum_{a:A} B_a))((Y_b)_b, \alpha^*((X_a)_a)).$$

This, again, suggests an adjunction which we will use as a definition.

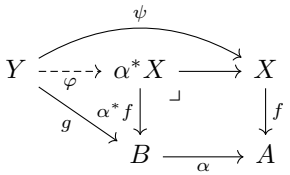
Definition 8. For a category C and a morphism $\alpha : B \to A$, the *dependent sum* along α is, if it exists, the left adjoint to the pullback functor:

$$(C \downarrow A) \underset{\alpha^*}{\overset{\sum_\alpha}{\underset{\longrightarrow}{\overset{\longleftarrow}{\bot}}}} (C \downarrow B)$$

However, note that the conversion from an external to an internal representation in **Set** already contained a dependent sum, which is no coincidence. It turns out that in practice, we will never have a hard time obtaining dependent sums:

LEMMA 2. *Let $\alpha : C(B, A)$ be a morphism in a category. If the pullback functor $\alpha^* : (C \downarrow A) \to (C \downarrow B)$ exists, it has a left adjoint given by postcomposition with α.*

PROOF. For morphisms $f : X \to A$, $g : Y \to B$, the universal property of the pullback, with the following diagram



gives an equivalence between morphisms $\varphi : Y \to \alpha^* X$ that commute with g and $\alpha^* f$, and morphisms $\psi : Y \to X$ that commute with g, f and α. In other words:

$$(C \downarrow A)(g \cdot \alpha, f) \simeq (C \downarrow B)(g, \alpha^*(f)),$$

which shows the adjunction. □

Now, let $f : X \to A$ be the internal representation of an indexed family $(X_a)_a$ and let $\alpha : A \to I$ be the terminal projection. We have $\sum_{a:A} X_a = f \circ \alpha : X \to I$. By the equivalence between $(C \downarrow I)$ and C, we see that the dependent sum of the family $(X_a)_a$ is exactly X. Therefore, our attention is mainly focused on the dependent product.

We will close this section with a name for a category that has all dependent products:

DEFINITION 9. A *locally cartesian closed* category is a category C with pullbacks such that each pullback functor α^* has a right adjoint.

Apart from having dependent sums and products, there also is the following theorem that shows the significance locally cartesian closedness:

LEMMA 3. *A category C is locally cartesian closed iff $(C \downarrow A)$ is cartesian closed for each $A : C$.*

PROOF. See the end of Section 1.3 of [**Fre72**]. □

7. (Weakly) terminal objects

DEFINITION 10. If a category has an object t, such that there is a (not necessarily unique) morphism to it from every other object in the category, t is said to be a *weakly terminal object*.

DEFINITION 11. Let C be a category with terminal object t. For an object $c : C$, a *global element* of c is a morphism $f : C(t, c)$.

8. Kan Extensions

One of the most general and abstract concepts in category theory is the concept of *Kan extensions*. In [**ML98**], Section X.7, MacLane notes that

> The notion of Kan extensions subsumes all the other fundamental concepts of category theory.

In this thesis, we will use left Kan extension a handful of times. It comes in handy when we want to extend a functor along another functor in the following way:

Let A, B and C be categories and let $F : A \to B$ be a functor.

DEFINITION 12. Precomposition gives a functor between functor categories $F_* : [B, C] \to [A, C]$. If F_* has a left adjoint, we will denote call this adjoint functor the *left Kan extension* along F and denote it $\mathrm{Lan}_F : [A, C] \to [B, C]$.

Analogously, when F_* has a right adjoint, one calls this the *right Kan extension* along F and denote it $\mathrm{Ran}_F : [A, C] \to [B, C]$.

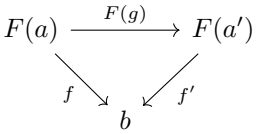
If a category has limits (resp. colimits), we can construct the right (resp. left) Kan extension in a 'pointwise' fashion (see Theorem X.3.1 in [**ML98**] or Theorem 2.3.3 in [**KS06**]). Below, I will outline the parts of the construction that we will need explicitly in this thesis.

LEMMA 4. *If C has colimits, Lan_F exists.*

PROOF. First of all, for objects $b : B$, we take

$$\mathrm{Lan}_F G(b) := \mathrm{colim}\left((F \downarrow b) \to A \xrightarrow{G} C\right).$$

Here, $(F \downarrow b)$ denotes the comma category with as objects the morphisms $B(F(a), b)$ for all $a : A$, and as morphisms from $f : B(F(a), b)$ to $f' : B(F(a'), b)$ the morphisms $g : A(a, a')$ that make the diagram commute:



and $(F \downarrow b) \to A$ denotes the projection functor that sends $f : B(F(a_1), b)$ to a_1.

Now, a morphism $h : B(b, b')$ gives a morphism of diagrams, sending the $F(a)$ corresponding to $f : B(G(a), b)$ to the $F(a)$ corresponding to $f \cdot h : B(G(a), b')$. From this, we get a morphism $\mathrm{Lan}_F G(h) : C(\mathrm{Lan}_F G(b), \mathrm{Lan}_F G(b'))$.

The unit of the adjunction is a natural transformation $\eta : \mathrm{id}_{[A,C]} \Rightarrow \mathrm{Lan}_F \bullet F_*$. We will define this pointwise, for $G : [A, C]$ and $a : A$. Our diagram contains the $G(a)$ corresponding to $\mathrm{id}_{F(a)} : (F \downarrow F(a))$ and the colimit cocone gives a morphism

$$\eta_G(a) : C(G(a), \mathrm{Lan}_F G(F(a))),$$

the latter being equal to $(\mathrm{Lan}_F \bullet F_*)(G)(a)$.

The counit of the adjunction is a natural transformation $\epsilon : F_* \bullet \mathrm{Lan}_F \Rightarrow \mathrm{id}_{[B,C]}$. We will also define this pointwise, for $G : [B, C]$ and $b : B$. The diagram for $\mathrm{Lan}_F(F_* G)(b)$ consists of $G(F(a))$ for all $f : B(F(a), b)$. Then, by the universal property of the colimit, the morphisms $G(f) : C(G(F(a)), G(b))$ induce a morphism

$$\epsilon_G(b) : C(\mathrm{Lan}_F(F_* G)(b), G(b)).$$

□

LEMMA 5. *If $F : A \to B$ is a fully faithful functor, and C is a category with colimits, $\eta : id_{[A,C]} \Rightarrow Lan_F \bullet F_*$ is a natural isomorphism.*

PROOF. To show that η is a natural isomorphism, we have to show that $\eta_G(a') : G(a') \Rightarrow \mathrm{Lan}_F G(F(a'))$ is an isomorphism for all $G : [A, C]$ and $a' : A$. Since a left adjoint is unique up to natural isomorphism ([**AW23**], Exercise 153), we can assume that $\mathrm{Lan}_F G(F(a'))$ is given by

$$\mathrm{colim}((F \downarrow F(a')) \to A \xrightarrow{G} C).$$

Now, the diagram for this colimit consists of $G(a)$ for each arrow $f : B(F(a), F(a'))$. Since F is fully faithful, we have $f = F(\overline{f})$ for some $\overline{f} : A(a, a')$. If we now take the arrows $G(\overline{f}) : C(G(a), G(a'))$, the universal property of the colimit gives an arrow

$$\varphi : C(\mathrm{Lan}_F G(F(a')), G(a'))$$

which constitutes an inverse to $\eta_G(a')$. The proof of this revolves around properties of the colimit and its (induced) morphisms. □

Remark 3. In the same way, if C has limits, ϵ is a natural isomorphism.

Corollary 1. *If C has limits or colimits, precomposition of functors $[B, C]$ along a fully faithful functor is (split) essentially surjective.*

Proof. For each $G : [A, C]$ we take $\mathrm{Lan}_F G : [B, C]$, and we have $F_*(\mathrm{Lan}_F G) \cong G$. □

Corollary 2. *If C has colimits (resp. limits), left (resp. right) Kan extension of functors $[A, C]$ along a fully faithful functor is fully faithful.*

Proof. Since left Kan extension along F is the left adjoint to precomposition, we have

$$[A, C](\mathrm{Lan}_F G, \mathrm{Lan}_F G') \cong [B, C](G, F_*(\mathrm{Lan}_F G')) \cong [B, C](G, G').$$

□

9. The Karoubi envelope

Let C be a category. If we have a retraction-section pair $c \underset{s}{\overset{r}{\rightleftarrows}} d$ we have (by definition) $s \cdot r = \mathrm{id}_d$. On the other hand, $r \cdot s : c \to c$ is an idempotent morphism, since $r \cdot s \cdot r \cdot s = r \cdot s$. Conversely, we can wonder whether for some idempotent morphism $a : c \to c$, we can find a retraction-section pair (r, s) such that $a = r \cdot s$. If this is the case, we say that the idempotent a *splits*. If a does not split, we can wonder whether we can find an embedding $\iota_C : C \hookrightarrow \overline{C}$ such that the idempotent $\iota_C(a) : \iota_C(c) \to \iota_C(c)$ does split. This is one way to arrive at the *Karoubi envelope*.

Definition 13. We define the category $\overline{C}$. The objects of $\overline{C}$ are tuples (c, a) with $c : C$, $a : C(c, c)$ such that $a \cdot a = a$. The morphisms between (c, a) and (d, b) are morphisms $f : C(a, b)$ such that $a \cdot f \cdot b = f$. The identity morphism on (c, a) is given by a and $\overline{C}$ inherits morphism composition from C.

This category is called the *Karoubi envelope*, the *idempotent completion*, the *category of retracts*, or the *Cauchy completion* of C.

Remark 4. Note that for a morphism $f : \overline{C}((c, a), (d, b))$,

$$a \cdot f = a \cdot a \cdot f \cdot b = a \cdot f \cdot b = f$$

and in the same way, $f \cdot b = f$.

Definition 14. We have an embedding $\iota_C : C \to \overline{C}$, sending $c : C$ to (c, id_c) and $f : C(c, d)$ to f.

Lemma 6. *Every object $c : \overline{C}$ is a retract of $\iota_C(c_0)$ for some $c_0 : C$.*

Proof. Note that $c = (c_0, a)$ for some $c_0 : C$ and an idempotent $a : c \to c$. We have morphisms $\iota_C(c) \underset{a_\leftarrow}{\overset{a_\rightarrow}{\rightleftarrows}} (c, a)$, both given by a. We have $a_\leftarrow \cdot a_\rightarrow = a = \mathrm{id}_{(c,a)}$, so (c, a) is a retract of $\iota_C(c)$. □

Lemma 7. *In $\overline{C}$, every idempotent splits.*

PROOF. Take an idempotent $e : \overline{C}(c, c)$. Note that c is given by an object $c_0 : C$ and an idempotent $a : C(c_0, c_0)$. Also, e is given by some idempotent $e : C(c_0, c_0)$ with $a \cdot e \cdot a = e$.

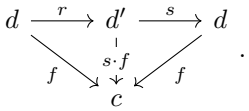
Now, we have $(c_0, e) : \overline{C}$ and morphisms $(c_0, a) \underset{e_\leftarrow}{\overset{e_\rightarrow}{\rightleftarrows}} (c_0, e)$, both given by e. We have $e_\leftarrow \cdot e_\rightarrow = e = \mathrm{id}_{(c_0,e)}$, so (c_0, e) is a retract of (c_0, a). Also, $e = e_\rightarrow \cdot e_\leftarrow$, so e is split. □

REMARK 5. Note that the embedding is fully faithful, since

$$\overline{C}((c, \mathrm{id}_c), (d, \mathrm{id}_d)) = \{f : C(c, d) \mid \mathrm{id}_c \cdot f \cdot \mathrm{id}_d = f\} = C(c, d).$$

REMARK 6. Let D be a category. Suppose that we have a retraction-section pair in D, given by $d \underset{s}{\overset{r}{\rightleftarrows}} d'$. Now, suppose that we have an object $c : D$ and a morphism f with $(r \cdot s) \cdot f = f$. Then we get a morphism $s \cdot f : d' \to c$ such that f factors as $r \cdot (s \cdot f)$. Also, for any g with $r \cdot g = f$, we have

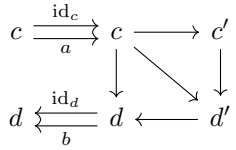
$$g = s \cdot r \cdot g = s \cdot f.$$



Therefore, d' is the equalizer of $d \underset{r \cdot s}{\overset{\mathrm{id}_d}{\rightrightarrows}} d$. In the same way, it is also the coequalizer of this diagram.

Now, note that if we have a coequalizer c' of id_c and a, and an equalizer d' of id_d and b, the universal properties of these give an equivalence

$$D(c', d') \cong \{f : D(c, d') \mid a \cdot f = f\} \cong \{f : D(c, d) \mid a \cdot f = f = f \cdot b\}.$$



Since a functor preserves retracts, and since every object of $\overline{C}$ is a retract of an object in C, one can lift a functor from C (to a category with (co)equalizers) to a functor on $\overline{C}$.

For convenience, the lemma below works with pointwise left Kan extension using colimits, but one could also prove this using just (co)equalizers (or right Kan extension using limits).

LEMMA 8. *Let D be a category with colimits. We have an adjoint equivalence between $[C, D]$ and $[\overline{C}, D]$.*

PROOF. We already have an adjunction $\mathrm{Lan}_{\iota_C} \dashv \iota_{C*}$. Also, since ι_C is fully faithful, we know that η is a natural isomorphism. Therefore, we only have to show that ϵ is a natural isomorphism. That is, we need to show that $\epsilon_G(c, a) : D(\mathrm{Lan}_{\iota_C}(\iota_{C*}G)(c, a), G(c, a))$ is an isomorphism for all $G : [\overline{C}, D]$ and $(c, a) : \overline{C}$.

One of the components in the diagram of $\mathrm{Lan}_{\iota_C}(\iota_{C*}G)(c, a)$ is the $\iota_{C*}G(c) = G(c, \mathrm{id}_c)$ corresponding to $a : \iota_C(c) \to (c, a)$. This component has a morphism into our colimit

$$\varphi : C(G(\iota_C(c)), \mathrm{Lan}_{\iota_C}(\iota_{C*}G)(c, a)).$$

Note that we can view a as a morphism $a : \overline{C}((c, a), \iota_C(c))$. This gives us our inverse morphism

$$G(a) \cdot \varphi : C(G(c, a), \mathrm{Lan}_{\iota_C}(\iota_{C*}G)(c, a)).$$

□

LEMMA 9. *The formation of the opposite category commutes with the formation of the Karoubi envelope.*

PROOF. An object in $\overline{C^{\mathrm{op}}}$ is an object $c : C^{\mathrm{op}}$ (which is just an object $c : C$), together with an idempotent morphism $a : C^{\mathrm{op}}(c,c) = C(c,c)$. This is the same as an object in $\overline{C}^{\mathrm{op}}$.

A morphism in $\overline{C^{\mathrm{op}}}((c,a),(d,b))$ is a morphism $f : C^{\mathrm{op}}(c,d) = C(d,c)$ such that

$$b \cdot_C f \cdot_C a = a \cdot_{C^{\mathrm{op}}} f \cdot_{C^{\mathrm{op}}} b = f.$$

A morphism in $\overline{C}^{\mathrm{op}}((c,a),(d,b)) = \overline{C}((d,b),(c,a))$ is a morphism $f : C(d,c)$ such that $b \cdot f \cdot a = f$.

Now, in both categories, the identity morphism on (c,a) is given by a.

Lastly, $\overline{C^{\mathrm{op}}}$ inherits morphism composition from C^{op}, which is the opposite of composition in C. On the other hand, composition in $\overline{C}^{\mathrm{op}}$ is the opposite of composition in $\overline{C}$, which inherits composition from C. □

COROLLARY 3. *As the category* **Set** *is cocomplete, we have an equivalence between the category of presheaves on C and the category of presheaves on $\overline{C}$:*

$$[C^{op}, \mathbf{Set}] \cong [\overline{C^{op}}, \mathbf{Set}] \cong [\overline{C}^{op}, \mathbf{Set}].$$

10. Monoids as categories

Take a monoid M.

DEFINITION 15. We can construct a category C_M with $C_{M0} = \{\star\}$, $C_M(\star,\star) = M$. The identity morphism on $\star$ is the identity $1 : M$. The composition is given by multiplication $g \cdot_{C_M} f = f \cdot_M g$.

REMARK 7. Actually, we have a functor from the category of monoids to the category of setcategories (categories whose object type is a set).

A monoid morphism $f : M \to M'$ is equivalent to a functor $F_f : C_M \to C_{M'}$. Any functor between C_M and $C_{M'}$ sends $\star_M$ to $\star_{M'}$. The monoid morphism manifests as $F_f(m) = f(m)$ for $m : C_M(\star,\star) = M$.

LEMMA 10. *An isomorphism of monoids gives an (adjoint) equivalence of categories.*

PROOF. Given an isomorphism $f : M \to M'$. Then we have functors $F_f : C_M \to C_{M'}$ and $F_{f^{-1}} : C_{M'} \to C_M$. Take the identity natural transformations $\eta : \mathrm{id}_{C_M} \Rightarrow F_f \bullet F_{f^{-1}}$ and $\epsilon : F_{f^{-1}} \bullet F_f \Rightarrow \mathrm{id}_{C_{M'}}$. Of course these are natural isomorphisms. □

DEFINITION 16. A *right monoid action* of M on a set X is a function $X \times M \to X$ such that for all $x : X$, $m, m' : M$,

$$x1 = x \qquad \text{and} \qquad (xm)m' = x(m \cdot m').$$

DEFINITION 17. A *morphism* between sets X and Y with a right M-action is an M-equivariant function $f : X \to Y$: a function such that $f(xm) = f(x)m$ for all $x : X$ and $m : M$.

These, together with the identity and composition from **Set**, constitute a category $\mathbf{RAct}_M$ of right M-actions.

LEMMA 11. *Presheaves on C_M are equivalent to sets with a right M-action.*

PROOF. This correspondence sends a presheaf F to the set $F(\star)$, and conversely, the set X to the presheaf F given by $F(\star) := X$. The M-action corresponds to the presheaf acting on morphisms as $xm = F(m)(x)$. A morphism (natural transformation) between presheaves $F \Rightarrow G$ corresponds to a function $F(\star) \to G(\star)$ that is M-equivariant, which is exactly a monoid action morphism. □

REMARK 8. Since the category of sets with an M-action is equivalent to a presheaf category, it has all limits. However, we can make this concrete. The set of the product $\prod_i X_i$ is the product of the underlying sets. The action is given pointwise by $(x_i)_i m = (x_i m)_i$.

Note that the initial set with M-action is $\{\star\}$, with action $\star m = \star$.

LEMMA 12. *The global elements of a set with right M-action correspond to the elements that are invariant under the M-action.*

PROOF. A global element of X is a morphism $\varphi : \{\star\} \to X$ such that for all $m : M$, $\varphi(\star)m = \varphi(\star m) = \varphi(\star)$. Therefore, it is given precisely by the element $\varphi(\star) : X$, which must be invariant under the M-action. □

LEMMA 13. *The category C of sets with an M-action has exponentials.*

PROOF. Given sets with M-action X and Y. Consider the set $C(M \times X, Y)$ with an M-action given by $\phi m'(m, x) = \phi(m'm, x)$. This is the exponential object X^Y, with the (universal) evaluation morphism $X \times X^Y \to Y$ given by $(x, \phi) \mapsto \phi(1, x)$. □

DEFINITION 18. We can view M as a set U_M with right M-action $mn = m \cdot_M n$ for $m : U_M$ and $n : M$.

10.1. Extension and restriction of scalars. Let $\varphi : M \to M'$ be a morphism of monoids.

Remember that sets with a right monoid action are equivalent to presheaves on the monoid category. Also, φ is equivalent to a functor between the monoid categories. The following is a specific case of the concepts in the section about Kan extension:

LEMMA 14. *We get a restriction of scalars functor φ_* from sets with a right M'-action to sets with a right M-action.*

PROOF. Given a set X with right M'-action, take the set X again, and give it a right M-action, sending (x, m) to $x\varphi(m)$.

On morphisms, send an M'-equivariant morphism $f : X \to X'$ to the M-equivariant morphism $f : X \to X'$. □

Since **Set** has colimits, and restriction of scalars corresponds to precomposition of presheaves (on $C_{M'}$), we can give it a left adjoint. This is the (pointwise) left Kan extension, which boils down to:

LEMMA 15. *We get an extension of scalars functor φ^* from sets with a right M-action to sets with a right M'-action.*

PROOF. Given a set X with right M-action. Take $Y = X \times M'/\sim$ with the relation $(xm, m') \sim (x, f(m) \cdot m')$ for $m : M$. This has a right M'-action given by $(x, m')n' = (x, m'n')$.

On morphisms, it sends the m-equivariant $f : X \to X'$ to the morphism $(x, m') \mapsto (f(x), m')$. □

LEMMA 16. *For U_M the set M with right M-action, we have $\varphi^*(U_M) \cong U_{M'}$.*

PROOF. The proof relies on the fact that for all $m : U_M$ and $m' : M'$, we have

$$(m, m') \sim (1, \varphi(m)m').$$

□

Consider the category D with $D_0 = M'$ and

$$D(m', \overline{m}') = \{m : M \mid \varphi(m) \cdot m' = \overline{m}'\}.$$

LEMMA 17. *Suppose that D has a weakly terminal element. Then for I_M the terminal object in the category of sets with a right M-action, we have $\varphi^*(I_M) \cong I_{M'}$.*

PROOF. If D has a weakly terminal object, there exists $m_0 : M'$ such that for all $m' : M'$, there exists $m : M$ such that $\varphi(m) \cdot m' = m_0$.

The proof relies on the fact that every element of $\varphi^*(I_M)$ is given by some $(\star, m')$, but then

$$(\star, m') = (\star \cdot m, m') \sim (\star, \varphi(m) \cdot m') = (\star, \overline{m'}),$$

so $\varphi^*(I_M)$ has exactly 1 element. □

REMARK 9. For φ^* to preserve terminal objects, we actually only need D to be connected. The fact that $\varphi^*(I_M)$ is a quotient by a symmetric and transitive relation then allows us to 'walk' from any $(\star, m'_1)$ to any other $(\star, m'_2)$ in small steps.

For any $m'_1, m'_2 : M'$, consider the category $D_{m'_1, m'_2}$, given by

$$D_{m'_1, m'_2, 0} = \{(m', m_1, m_2) : M' \times M \times M \mid m'_i = \varphi(m_i) \cdot m'\}$$

and

$$D_{m'_1, m'_2}((m', m_1, m_2), (\overline{m}', \overline{m}_1, \overline{m}_2)) = \{m : M \mid \varphi(m) \cdot m' = \overline{m}', m_i = \overline{m}_i \cdot m\}.$$

LEMMA 18. *Suppose that $D_{m'_1, m'_2}$ has a weakly terminal object for all $m'_1, m'_2 : M'$. Then for sets A and B with right M-action, we have $\varphi^*(A \times B) \cong \varphi^*(A) \times \varphi^*(B)$.*

PROOF. Now, any element in $\varphi^*(A) \times \varphi^*(B) = (A \times M'/\sim) \times (B \times M'/\sim)$ is given by some (a, m'_1, b, m'_2).

The fact that $D_{m'_1, m'_2}$ has a weakly terminal object means that we have some $\overline{m}' : M'$ and $\overline{m}_1, \overline{m}_2 : M$ with $m'_i = \varphi(\overline{m}_i) \cdot \overline{m}'$. Therefore,

$$(a, m'_1, b, m'_2) = (a, \varphi(\overline{m}_1) \cdot \overline{m}', b, \varphi(\overline{m}_2) \cdot \overline{m}') \sim (a\overline{m}_1, \overline{m}', b\overline{m}_2, \overline{m}'),$$

so this is equivalent to some element in $\varphi^*(A \times B) = (A \times B \times M'/\sim)$. Note that this trivially respects the right M'-action.

The fact that $(\overline{m}', \overline{m}_1, \overline{m}_2)$ is weakly terminal also means that for all $m' : M'$ and $m_1, m_2 : M$ with $m'_i = \varphi(m_i) \cdot m'$, there exists $m : M$ such that $\varphi(m) \cdot m' = \overline{m}'$ and $m_i = \overline{m}_i \cdot m$. This means that the equivalence that we established is actually well-defined: equivalent elements in $\varphi^*(A) \times \varphi^*(B)$ are sent to equivalent elements in $\varphi^*(A \times B)$.

Therefore, we have an isomorphism $\psi : \varphi^*(A) \times \varphi^*(B) \xrightarrow{\sim} \varphi^*(A \times B)$. Now we only need to show that the projections are preserved by this isomorphism. To that end, take $x = (a, m'_1, b, m'_2) \sim (a\overline{m}_1, \overline{m}', b\overline{m}_2, \overline{m}') : \varphi^*(A) \times \varphi^*(B)$. We have

$$\varphi^*(\pi_1)(\psi(x)) = (a\overline{m}_1, \overline{m}') = \pi'_1(x).$$

In the same way, $\varphi^*(\pi_2) \circ \psi = \pi'_2$ and this concludes the proof. □

CHAPTER 2

Univalent Foundations

1. Dependent type theory

Univalent foundations takes place in a framework of type theory. In this section, we will quickly introduce some topics that we will need in subsequent sections.

First of all, *type theory* is the study of 'type systems'. A *type system* is a collection of terms or elements, each of which has a corresponding type. A type is much like a set in set theory in that it has elements (for example, true : Bool or $1 : \mathbb{N}$), but there are important differences. First of all, in type theory, elements are declared together with their type, whereas in set theory, something can be an element of multiple sets, like $5 \in \mathbb{N}$ and $5 \in \mathbb{R}$. Secondly, equality in type theory can work a bit differently than in classical mathematics, but will cover this in the next section.

One notable class of types is given by the *function types*. For types A and B, our type system might have the type $A \to B$. An element f of this type can be combined with an element a of A to give an element $f(b)$ of B. Of course, the elements of this type are thought of (and usually are) functions from A to B.

Now, a type system may or may not have all kinds of constructs. One of these constructs is *dependent types*. A dependent type is a type which depends on values of other types. For example, $\texttt{array}(T, n)$, the type of arrays of length n, with elements of type T. The study of type systems that have such dependent types is called *dependent type theory*.

Suppose that we have a type A, and a dependent type which we will write $B : A \to \mathbf{Type}$.

One of the possible constructs in a type system is a type $\sum_{a:A} B(a)$ called the *dependent sum*, consisting of all pairs (a, b) with $a : A$ and $b : B(a)$. So every element of $\sum_{a:A} B(a)$ gives an element of one $B(a)$ (for some $a : A$). Note that for the constant dependent type $B(a) = B$, $\sum_{a:A} B$ is the product type $A \times B$.

Another construct which may exist, is a type $\prod_{a:A} B(a)$, consisting of all 'functions' f which map elements $a : A$ to elements $f(a) : B(a)$. Every element of $\prod_{a:A} B(a)$ gives therefore elements of all the $B(a)$ simultaneously. Note that $\prod_{a:A} B$ is the function type $A \to B$.

Lastly, there is a very strong connection between logic and type theory. This is called the *Curry-Howard correspondence* or sometimes referred to as *products as types*. We can view a type T as the proposition "T has an element". An element of T is then a 'proof' or 'witness' of this proposition. If the type system is strong enough, it allows us to do mathematics in it, where:

- A function $f : A \to B$ that takes an argument of type A and yields a result of type B corresponds to the proof of B under the hypothesis that A holds: "suppose that A, then B". Note that the notation $A \to B$ also makes sense if we think of A and B propositions.
- The empty type $\mathbf{0}$ corresponds to 'false'. Note that for all types A, we can construct a function $\mathbf{0} \to A$. In other words, we can prove everything from 'false'.

- The unit type $\mathbf{1}$ corresponds to 'true'.
- The negation of T is the function type $T \to \mathbf{0}$.
- The conjuction "A and B" is given as the product type $A \times B$.
- The disjunction "A or B" is given as the propositional truncation of the coproduct, union or sum type $A \sqcup B$.
- The statement "For all $a : A$, $B(a)$ holds", for some dependent type B (i.e. predicate) over A, is given as the dependent product $\prod_{a:A} B(a)$.

If we think of types as propositions, the question whether some axiom is assumed or not becomes the question whether some (family of) type(s) is inhabited. Note that most of the axioms that we list here require the notion of mere propositions, propositional truncation and mere existence, which we will cover from Section 2.4 onwards.

- If for all mere propositions A, the type $\|A \vee (A \to \mathbf{0})\|$ is inhabited, we say that the type system assumes the *axiom of excluded middle*: "Either A is true, or A is not true". This is axiom allows us to prove A from 'not not A'.
- If for all dependent types C over A and B, the type
$$\left(\prod_{a:A} \exists(b : B), C(a, b)\right) \to \exists(f : A \to B), \prod_{a:A} C(a, f(a))$$
is inhabited, we say that the type system assumes the *propositional axiom of choice*: "The product of a family of nonempty sets is nonempty".
- In any type theory which has the required constructs, the following holds: For all dependent types C over A and B, the type
$$\left(\prod_{a:A} \sum_{b:B} C(a, b)\right) \to \sum_{f:A\to B} \prod_{a:A} C(a, f(a))$$
is inhabited. This axiom (or actually more of a theorem) is called the *type theoretic axiom of choice*.

2. The univalence principle

Isomorphic objects are equivalent.

This principle is visible in most of mathematics: Sets with a bijection have the same number of elements, isomorphic groups have the same properties, and since the universal property of limits makes them "unique up to unique isomorphism", we can talk about 'the' limit of some diagram in a category.

Now, the *univalence principle* takes this a step further. It states that

Isomorphic objects are equal.

Univalent foundations seeks to be a foundation for mathematics that is in line with this principle. This is often done within the framework of 'Martin-Löf dependent type theory', a type theory developed by Per Martin-Löf [**ML71**]. It is a family of dependent type systems with dependent products, dependent sums, an empty type, a unit type and union types. It is a constructive type theory, so it does not automatically assume the axiom of excluded middle or the propositional axiom of choice. It is however compatible with these axioms, so one can still assume these alongside its usual axioms.

It is important to note that Martin-Löf type theory has *identity types*: given a type T and elements $x, y : T$, we have a type $\mathrm{Id}_T(x, y)$ (note that this is a dependent type), which we will usually denote with $x = y$. An element $p : x = y$ is a proof that x is 'equal' to y. This type comes with an interesting induction principle called *path induction*: we can show any statement about paths $p : x = y$ for generic x

and y, if we can show it about the 'trivial' path $\mathrm{refl} : x = x$ for generic x. For example, symmetry of the equality boils down to a function $\prod_{xy:T} x = y \to y = x$. We construct this using path induction with the function that sends $\mathrm{refl} : x = x$ to itself. For more information, see [**Uni13**], Section 1.12.1.

In set-theoretic mathematics, there is the concept of a 'bijection' $S \simeq T$ of sets (or an isomorphism in the category **Set**), which is often treated as an equivalence. It consists of functions $f : S \to T$ and $g : T \to S$ with $f \cdot g = \mathrm{id}_S$ and $g \cdot f = \mathrm{id}_T$. In type theory, we have a similar concept, which is called 'equivalence' (of types) $S \simeq T$. Since bijections are not well-behaved for types that are not sets (see 2.4), because in those cases, a function $f : S \to T$ can have multiple distinct inverses. Therefore, we define $S \simeq T := \sum_{f:S\to T} \mathrm{isequiv}(f)$, for some predicate $\mathrm{isequiv} : (S \to T) \to \mathbf{Type}$ (see [**Uni13**], Equation 2.4.10). However, intuitively we can still think of these as bijections.

Using the identity type, we can make our statement of the univalence principle more precise (and a bit stronger). For types, we can construct a function

$$\mathrm{idtoequiv} : \prod_{S,T:\mathbf{Type}} (S = T) \to (S \simeq T).$$

We construct this function using path induction with the identity bijection $\prod_S \mathrm{id}_S : (S \simeq S)$. In fact, this is a specific case of the following: for a category C, if we denote the type of isomorphisms between objects c and d with $c \cong d$, we can construct a function

$$\mathrm{idtoiso} : \prod_{c,d:C} (c = d) \to (c \cong d),$$

using path induction with the identity isomorphism $\prod_{c:C} \mathrm{id}_c : (c \cong c)$. We can then formulate the univalence principle for categories as

For all $c, d : C$, $\mathrm{idtoiso}_{c,d} : (c = d) \to (c \cong d)$ is an equivalence.

A category that adheres to the univalence principle is called a *univalent category*.

3. The univalence axiom

Now, even for a basic category, like the category of types, it seems impossible to prove that the univalence principle holds. However, this is no surprise: it turns out that it is independent of the axioms of Martin-Löf type theory (**(TODO) Is this true? I need a reference for this, but could not find one**). That is: it cannot be proven, but assuming it as an axiom does not yield contradictions.

As mentioned before, univalent foundations attempts to develop as much of mathematics as possible along the univalence principle. Therefore, we assume as our first axiom the *univalence axiom*:

Axiom. *For all $S, T :$ **Type**, the function $\mathrm{idtoequiv}_{S,T} : (S = T) \to (S \simeq T)$ is an equivalence.*

In other words:

Axiom. **Type** *is univalent.*

Remark 10. One consequence of the independence of the univalence axiom is that equivalent objects are 'indiscernible'. That is: even if we do not yet assume the univalence axiom, we cannot formulate a property that is satisfied by some type, but not by another, equivalent type. This is because such a property would yield a contradiction when we would assume the univalence axiom.

Now, the question arises: how about the univalence axiom for categories other than **Type**? Do we need to keep assuming an additional axiom for every category that we want to be univalent? It turns out that this is not necessary. In practice,

most categories consist of 'sets (or types) with additional structure'. For example: topological spaces, groups, λ-theories and algebraic theory algebras. In such categories, we can leverage the univalence of **Type** to show that for isomorphic objects, their underlying types are equal. Also morphisms are usually defined in such a way that they 'preserve' the 'additional structure', which is what we need to show that the category is univalent.

Also, Theorem 4.5 in [**AKS15**] shows that if a category B is univalent (in the paper, categories are called 'precategories' and univalent categories are just called 'categories'), then the functor category $A \to B$ is also univalent. In particular, the category of (pre)sheaves $A \to \mathbf{Set}$ is univalent.

Therefore, the univalence axiom is a very powerful axiom, and we usually do not need to assume additional axioms to show that more categories satisfy the univalence principle.

The last structure in this section for which we want to consider the univalence axiom, is the 2-category **Cat** of categories. In general, we cannot show that this satisfies the univalence principle. However, we will restrict our attention to the sub-2-category of univalent categories, which are the categories that we want to study. Then Theorem 6.8 in [**AKS15**] shows that for univalent categories C and D, there is an equivalence between $C = D$ and $C \simeq D$, where $C \simeq D$ denotes the type of (adjoint) equivalences of categories (see Definition 2).

Lastly, a result about univalent categories that we will use a couple of times in this thesis:

LEMMA 19. *For a functor between univalent categories $F : A \to B$, the types 'F is an adjoint equivalence' and 'F is fully faithful and essentially surjective' are equivalent propositions (see 2.4).*

PROOF. See [**AKS15**], Lemma 6.8. □

4. hProps and hSets

If we have a type T and objects x and y, we can wonder how many elements $x = y$ has. In set-based mathematics, this would be a nonsensical question: two elements of a set are either equal or not equal. Therefore, we can expect the answer to be that $x = y$ has at most one element. And indeed, if we do not assume the univalence axiom, we can assume another axiom, called 'uniqueness of identity proofs', which states that for $p, q : x = y$, we have a proof of equality $h : p = q$.

On the other hand, suppose that we do assume the univalence axiom. Consider the two-element type $T = \{-1, 1\}$. We can construct two different equivalences $\mathrm{id}_T, \sigma : T \simeq T$:

$$\mathrm{id}_T(x) = x \quad \text{and} \quad \sigma(x) = -x.$$

By the univalence axiom, we must have that the identity type $(T = T)$ has (at least) two distinct elements, corresponding to id_T and σ. Therefore, the univalence axiom is not compatible with uniqueness of identity proofs, and we see that in a univalent setting, some identity types have more than one element.

A consequence of this is that types in general have too little structure to serve as a foundation for mathematics that was originally set-based. For example, suppose that we want to formalize the theory of groups. A group homomorphism $f : \mathbf{Grp}(H, G)$ is defined as a function on the underlying types $f_1 : H \to G$, together with a proof that it commutes with the group operations: $f_2 : \prod_{x,y:H} f_1(x \circ y) = f_1(x) \circ f_1(y)$. Now, normally in group theory, to show that two homomorphisms $f, g : \mathbf{Grp}(H, G)$ are equal, we show that $\prod_{x:H} f_1(x) = g_1(x)$, the 'data' is equal. However, if we are working with types instead of sets, we also need to show that the proofs of the 'properties' are equal: $f_2 = g_2$ (note that we

actually would need to transport f_2 here, for this equality to typecheck). This makes showing equality of morphisms much more complicated for concrete groups, and sometimes outright impossible for generic groups.

To deal with this, we need the concepts of mere propositions and (homotopy) sets:

DEFINITION 19. A *mere proposition* is a type T such that for all $x, y : T$, $x = y$.

DEFINITION 20. A *homotopy set* is a type T such that for all $x, y : T$, $x = y$ is a mere proposition.

Since the identity types for a homotopy set are mere propositions, a homotopy set mimicks a set in set theory, where equality between elements 'is' or 'is not'. If we restrict the underlying type of a group to be a homotopy set, it can be shown that $\prod_{x,y:H} f_1(x \circ y) = f_1(x) \circ f_1(y)$ is a mere proposition, so $f_2 = g_2$ trivially. This is often true when translating definitions from set theory to univalent foundations: if we base our objects on homotopy sets instead of types, we only have to worry about equality of 'data', the equality of 'properties' follows automatically.

For similar reasons, we restrict the hom-types $C(c, d)$ of categories to be sets. Note that **Type** is not a category under this definition (it is a 'precategory', for some definition of precategory), because in general, the type of functions between sets $S \to T$ is not a set.

5. truncation and (mere) existence

As mentioned before, if we want to do mathematics in a dependent type theory, we can 'encode' propositions as types. The elements of the type correspond to the proofs that the proposition is true, see for example the identity types. However, we need to be careful here about the distinction between types in general and mere propositions:

A proof that a type T is nonempty usually consists of giving an element $t : T$. If we encode the statement "T is nonempty" as T, and if T is not a mere proposition, then "T is nonempty" is not a mere proposition, so it has multiple distinct proofs. In some cases, this is exactly what we want, because we want to retrieve the chosen element of T. However, in some cases, we want the fact that a set is nonempty (or any statement in general) to be a mere proposition, to avoid having to carry around a specific element and having to prove equality of two specific elements. For such cases, we have the 'propositional truncation':

DEFINITION 21. For a type T, a type $\|T\|$ exists ([**Uni13**], Section 3.7) with the properties that for all $t : T$, we have an element $|t| : \|T\|$ and that $\|T\|$ is a mere proposition. It has a recursion principle stating that for a mere proposition B, a function $f : A \to B$ induces a function $|f| : \|T\| \to B$ that commutes with $|\cdot|$. This object is called the *propositional truncation*.

The propositional truncation forgets the details about a type, and only keeps the information about whether it is inhabited or not. The recursion principle means that if we are trying to prove a mere proposition based on some element $|t| : \|T\|$, we can pretend that we do have a concrete element $t : T$.

There also is the concept of higher order truncations. For example, the *set truncation* $\|A\|_0$, which is a homotopy set and has an equivalence $(\|A\|_0 \to B) \simeq (A \to B)$ for any set B. However, these higher truncation types become increasingly harder to construct, and in this thesis, we will only need to consider the propositional truncation.

Often, when we prove a theorem or lemma that "there exists some $x : X$ such that $Y(x)$", what we actually mean is that we can construct an element $x : X$ and

then an element $y : Y(x)$ of the dependent type Y over X. This is equivalent to having an element of $\sum_{x:X} Y(x)$. However, this is in general of course not a mere proposition. If we want to express that the set of such x is nonempty as a mere proposition, we talk about *mere existence*:

DEFINITION 22. Given a dependent type Y over X, if we say that there *merely exists* an element $x : X$ such that $Y(x)$, we mean that we have an element of the propositional truncation

$$h : (\exists x : Y(x)) := \left\| \sum_{x:X} Y(x) \right\|.$$

For example, if we have objects in a category $c, d : C$ and we want to talk about a retraction f of c onto d, we commonly define this as "a morphism $f_1 : C(c, d)$, such that there exists a 'section': a morphism $f_2 : C(d, c)$ with $f_2 \cdot f_1 = \mathrm{id}_d$". Now, we commonly consider f_1 to be the 'data' of the retraction; we consider retractions f and f' to be the same retraction if $f_1 = f'_1$. This means that being a retraction is about the 'mere existence' of a section. Note, however, that in this case, we cannot use the section in constructions, except when we are trying to prove mere propositions.

Note that for the Curry-Howard correspondence, the product or conjunction $A \wedge B = A \times B$ of two mere propositions is again a mere proposition. However, the union $A \sqcup B$ is not necessarily a mere proposition. To make it into a mere proposition, we need to take the propositional truncation $A \vee B = \|A \sqcup B\|$.

6. Equality and homotopy

Univalent Foundations is often mentioned together with Homotopy Type Theory, because they are related but distinct concepts. Therefore, we will mention it here.

As we saw before, if we do not assume uniqueness of identity proofs, given two elements $x, y : T$, we can have multiple distinct proofs that $x = y$, which is counterintuitive. One way to think about this, is the perspective of homotopy type theory. In homotopy type theory, one considers a type T to be a 'space', intuitively similar to a topological space, but without an explicit topology. The elements $t : T$ are then the points of the space. Elements of the identity type $(s = t)$ are interpreted as 'paths' from s to t. That is why the induction principle of $(s = t)$ is called 'path induction'. Of course, we can go higher: for $p, q : x = y$, the elements $(p = q)$ are 'paths between paths', (path) 'homotopies', 'sheets' or '1-cells', for homotopies $h, h' : p = q$, the elements of $h = h'$ are paths between paths between paths, 'volumes' or '2-cells' etc.

If we have a 'geometric' interpretation of our type theory, we can investigate the 'shape' of a (nonempty) type T, given by the structure of the (higher) identity types.

First of all, if we have $x, y : T$ for which $x = y$ does not have an inhabitant, x and y lie in different 'connected components'. Now, we focus on a connected type:

For example, are all elements $x, y : T$ of the type equal to each other, and are all elements $p, q : x = y$ of all the (higher) identity types also equal in a unique way? Then we have a *contractible type*, which intuitively looks somewhat like a plane (Figure 1a).

It is also possible that any two elements $x, y : T$ are equal, but that there are distinct paths $p, q : x = y$, with no homotopy between them. Then intuitively T looks like a circle or a tube or a projective space, or something more complicated (Figure 1b).

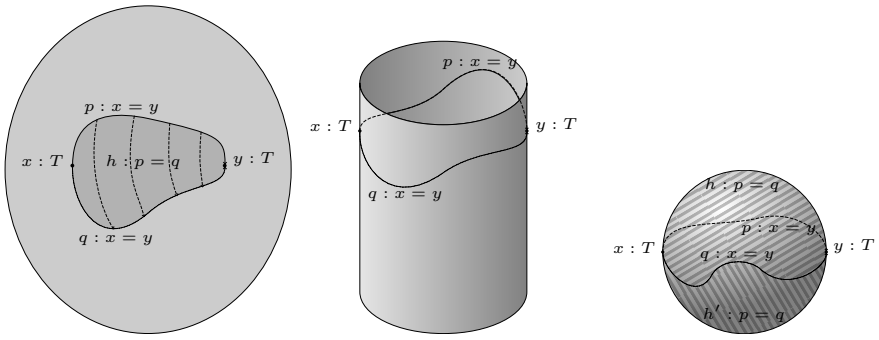


FIGURE 1. Different possible homotopy structures

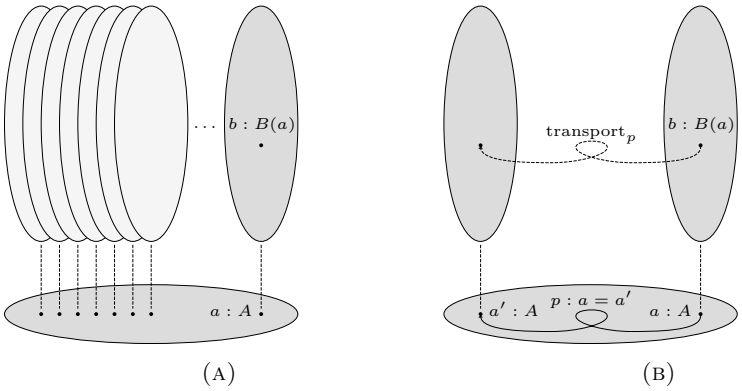


FIGURE 2. A fibration with a path in the base space, giving rise to transport in the fibration

REMARK 11. Note that we can give the type of paths $x = x$ a group structure and $x = y$ is a 'torsor' for this group. If $x = y$ has exactly two distinct elements, the group $x = x$ looks like $\mathbb{Z}/2\mathbb{Z}$, which suggests some projective plane-like structure.

For something to look like a circle or tube, we need this group $x = x$ to be isomorphic to $\mathbb{Z}$. For an example, see [**BBGS20**].

As a third example, consider a type T in which any two elements $x, y : T$ are equal, and any two paths $p, q : x = y$ have two distinct homotopies $h, h' : p = q$ between them. Then we can imagine the type to look somewhat like a sphere (Figure 1c) or something more complicated.

This is the lens through which homotopy type theory studies types.

7. Transport

Suppose that we have a dependent type $B : A \to UU$. Intuitively, B is a collection of spaces, lying over the points of A (Figure 2a). Now, if we have a path $p : a = a'$ in A, we can use path induction to get a function $\mathrm{transport}_p : B(a) \to B(a')$ (Figure 2b).

For example, this allows us to transfer an element from $\mathtt{array}(T, n + n)$ to $\mathtt{array}(T, 2 \cdot n)$ with transport over the path $n + n = 2 \cdot n$. Also, if we assume the univalence axiom, and we have, for example, an isomorphism of groups $f : G \cong G'$ and a proof h that G is semisimple, we can transport h along the equality given by f to a proof that G' is semisimple.

EXAMPLE 1. In addition, consider the following example. We take A to be a type universe, and let $B : T \mapsto \mathtt{array}(T, 3)$. Consider the types $a = \{\top, \bot\}$ and $a' = \{0, 1\}$. We have equivalences $\overline{p}$ and $\overline{q}$ between a and a':

$$\overline{p}(\top) = 1, \overline{p}(\bot) = 0 \quad \text{and} \quad \overline{q}(\top) = 0, \overline{q}(\bot) = 1.$$

That means that we have distinct equalities $p, q : a = a'$. Now, suppose that we have an array $x = [\top, \top, \bot] : B(a)$. Since $a = a'$, we would want to treat x as an element of $B(a')$, but then we need to make a choice whether we treat it as $[1, 1, 0]$ or $[0, 0, 1]$ (or something else altogether). This is exactly the question whether we transport along p or q, and therefore, when our base type is not a set, it is important to be aware that we are transporting a property along an equality, and we need to be careful which equality it is that we transport along.

Another place where transports occur frequently is when proving equality $(x, y) = (x', y')$ of elements of a dependent sum $\sum_{a:A} B(a)$. We can start componentwise by proving $p : x = x'$, but after this, we cannot directly prove $q : y = y'$, because these two live in different types: $B(x)$ and $B(x')$ respectively. Therefore, we need to transport, and then prove

$$\mathrm{transport}_p(y) = y' : B(x').$$

7.1. Caution: 'transport hell'. Now, it seems that we can transport all properties and data along equalities. And of course, that is true, but some caution needs to be taken with this when working with a proof assistant.

For example, consider the situation in Example 1. As mentioned, we can transport x to get an element $y := \mathrm{transport}_p(x)$, which is an element of $B(a')$, but now suppose that we want to compute something using its first coefficient y_1. Of course, on paper it quickly becomes clear that $y_1 = 1$, but in practice, it takes quite some work to convince a proof assistant of that fact. Of course, we could write a lemma which states that for any array x, any equivalence $\overline{p}$ with corresponding equality p, and any index i,

$$\mathrm{transport}_p(x)_i = \overline{p}(x_i).$$

However, at that point, we have more or less constructed our own function between $B(a)$ and $B(a')$, which is much easier to work with than $\mathrm{transport}_p$.

Experience teaches that in general, it is fine to transport properties $b : B(a)$ of which we will never need the value, only the fact that it (merely) exists. On the other hand, for properties and constructions of which we might later want to use the actual value, it is much better to transfer them 'by hand'. Often, but not always, this criterion coincides with $B(a)$ being a mere proposition.

Another case where unwanted transports often occur is when showing the equality of two complicated elements of a type. For example, $((x_1, x_2), (x_3, x_4)) = ((x_1', x_2'), (x_3', x_4'))$, both elements of

$$\sum_{(x_1,x_2):\sum_{a:A} B(a)} \sum_{x_3:C(x_1)} D(x_1, x_2, x_3).$$

This example involves proofs:

- $p : x_1 = x_1'$;
- $q : \mathrm{transport}_p(x_2) = x_2'$;
- $r : \mathrm{transport}_{(p,q)}(x_3) = x_3'$, which can be simplified to $\mathrm{transportf}_p(x_3) = x_3'$, since x_3 does not depend on x_2
- and finally $s : \mathrm{transport}_{((p,q),r)}(x_4) = x_4'$.

In general, these latter proofs r and s are very challenging and best avoided whenever possible. The situation in which one has to work with such complicated transports, which make a proof (seemingly unnecesarily) complicated is called *transport*

hell. Some mathematical tools have been developed (see for example 6.3 about displayed categories) that help avoid transport hell in some cases. Additionally, the structure or 'strategy' of a proof can sometimes be changed in order to avoid the worst of the transports. However, it is often not possible to avoid transports altogether, and this is one of the reasons why proving something in a proof assistant takes a lot more work than proving it on paper.

CHAPTER 3

Algebraic Structures

1. Algebraic Theories

DEFINITION 23. We define an *algebraic theory* T to be a sequence of sets T_n indexed over $\mathbb{N}$ with for all $1 \leq i \leq n$ elements ("variables" or "projections") $x_{n,i} : T_n$ (we usually leave n implicit), together with a substitution operation

$$\_ \bullet \_ : T_m \times T_n^m \to T_n$$

for all m, n, such that

$$\begin{aligned} x_j \bullet g &= g_j \\ f \bullet (x_{l,i})_i &= f \\ (f \bullet g) \bullet h &= f \bullet (g_i \bullet h)_i \end{aligned}$$

for all $1 \leq j \leq l$, $f : T_l$, $g : T_m^l$ and $h : T_n^m$.

DEFINITION 24. A *morphism* F between algebraic theories T and T' is a sequence of functions $F_n : T_n \to T'_n$ (we usually leave the n implicit) such that

$$\begin{aligned} F_n(x_j) &= x_j \\ F_n(f \bullet g) &= F_m(f) \bullet (F_n(g_i))_i \end{aligned}$$

for all $1 \leq j \leq n$, $f : T_m$ and $g : T_n^m$.

Together, these form the category of algebraic theories **AlgTh**.

REMARK 12. We can construct binary products of algebraic theories, with sets $(T \times T')_n = T_n \times T'_n$, variables (x_i, x_i) and substitution

$$(f, f') \bullet (g, g') = (f \bullet g, f' \bullet g').$$

In the same way, the category of algebraic theories has all limits.

Later on, we will see an example of a trivial algebraic theory (the terminal theory) T, in which every T_n only contains one element. However, we can say a bit about nontrivial algebraic theories.

LEMMA 20. *Let T be an algebraic theory, such that T_n has at least two distinct elements for some n. Then for all $1 \leq i, j \leq m$ with $i \neq j$, we have $x_{m,i} \neq x_{m,j}$.*

PROOF. We can also formulate the statement as: If there exist $1 \leq i, j \leq m$ with $i \neq j$ such that $x_{m,i} = x_{m,j}$, then all L_n contain at most one distinct element.

So, suppose that $x_{m,i} = x_{m,j}$ for some $i \neq j$. For $a, b : L_n$, we define $v : L_n^m$ as

$$v_k = \begin{cases} a & k = i \\ b & k \neq i \end{cases},$$

so in particular, $v_j = b$. Then we have

$$a = v_i = x_{m,i} \bullet v = x_{m,j} \bullet v = v_j = b,$$

so L_n contains at most one distinct element. □

2. Algebras

DEFINITION 25. An *algebra* A for an algebraic theory T is a set A, together with an action

$$\bullet : T_n \times A^n \to A$$

for all n, such that

$$x_j \bullet a = a_j$$
$$(f \bullet g) \bullet a = f \bullet (g_i \bullet a)_i$$

for all j, $f : T_m$, $g : T_n^m$ and $a : A^n$.

DEFINITION 26. For an algebraic theory T, a *morphism* F between T-algebras A and A' is a function $F : A \to A$ such that

$$F(f \bullet a) = f \bullet (F(a_i))_i$$

for all $f : T_n$ and $a : A^n$.

Together, these form the category of T-algebras $\mathbf{Alg}_T$.

REMARK 13. The category of algebras has all limits. The set of a limit of algebras is the limit of the underlying sets.

REMARK 14. Note that for an algebraic theory T, the T_n are all algebras for T, with the action given by $\bullet$.

DEFINITION 27 (Pullback of algebras). If we have a morphism of algebraic theories $f : T' \to T$, we have a functor $\mathbf{Alg}_T \to \mathbf{Alg}'_T$. It endows T'-algebras with an action from T given by $g \bullet_{T'} a = f(g) \bullet_T a$. Then T'-algebra morphisms commute with this T-action, so we indeed have a functor.

We can also consider the category of 'all' algebraic theory algebras together. That is, the category C with $C_0 = \sum_{T:\mathbf{AlgTh}}, \mathbf{Alg}_T$ and $C((T, A), (T', A'))$ consisting of pairs $(f, f') : \mathbf{AlgTh}(T, T') \times \mathbf{Set}(A, A')$ such that for all $t : T_n$ and $a : A^n$,

$$f'(t \bullet a) = f(t) \bullet (f'(a_i))_i.$$

We then have a functor $P : C \to \mathbf{AlgTh}$, projecting onto the first coordinate.

LEMMA 21. *P is a fibration.*

PROOF. Given an algebraic theory morphism $f : \mathbf{AlgTh}(S, T)$ and a T-algebra A_T, Definition 27 gives an S-algebra A_S with underlying set A_T. The cartesian morphism is $(f, \mathrm{id}_{A_T}) : C((S, A_S), (T, A_T))$.

It is cartesian because for $(R, A_R) : C$ and $(g, g') : C((R, A_R), (T, A_T))$ and $h : \mathbf{AlgTh}(R, S)$ with $h \cdot f = g$, the required morphism $C((R, A_R), (S, A_S))$ is given by $g' : \mathbf{Set}(A_R, A_S)$. □

3. Presheaves

DEFINITION 28. A *presheaf* P for an algebraic theory T is a sequence of sets P_n indexed over $\mathbb{N}$, together with an action

$$\bullet : P_m \times T_n^m \to P_n$$

for all m, n, such that

$$t \bullet (x_{l,i})_i = t$$
$$(t \bullet f) \bullet g = t \bullet (f_i \bullet g)_i$$

for all $t : P_l$, $f : T_m^l$ and $g : T_n^m$.

DEFINITION 29. For an algebraic theory T, a *morphism* F between T-presheaves P and P' is a sequence of functions $F_n : P_n \to P'_n$ such that

$$F_n(t \bullet f) = F_m(t) \bullet f$$

for all $t : P_m$ and $f : T_n^m$.

Together, these form the category of T-presheaves $\mathbf{Pshf}_T$.

REMARK 15. The category of presheaves has all limits. The nth set $\overline{P}_n$ of a limit $\overline{P}$ of presheaves P_i is the limit of the nth sets $P_{i,n}$ of the presheaves in the limit diagram.

An analogue to Lemma 21 shows that, like with algebras, the total category of presheaves is fibered over the category of algebraic theories.

4. λ-theories

Let $\iota_{m,n} : T_m \to T_{m+n}$ be the function that sends f to $f \bullet (x_{m+n,1}, \ldots, x_{m+n,m})$. Note that

$$\iota_{m,n}(f) \bullet g = f \bullet (g_i)_{i \leq m} \quad \text{and} \quad \iota_{m,n}(f \bullet g) = f \bullet (\iota_{m,n}(g_i))_i.$$

For tuples $x : X^m$ and $y : X^n$, let $x{+}y$ denote the tuple $(x_1, \ldots, x_m, y_1, \ldots, y_n) : X^{m+n}$.

DEFINITION 30 (λ-theory). A *λ-theory* is an algebraic theory L, together with sequences of functions $\lambda_n : L_{n+1} \to L_n$ and $\rho_n : L_n \to L_{n+1}$, such that

$$\lambda_m(f) \bullet h = \lambda_n(f \bullet ((\iota_{n,1}(h_i))_i + (x_{n+1})))$$
$$\rho_n(g \bullet h) = \rho_m(g) \bullet ((\iota_{n,1}(h_i))_i + (x_{n+1}))$$

for all $f : L_{m+1}$, $g : L_m$ and $h : L_n^m$.

Together, these form the category of λ-theories **LamTh**.

DEFINITION 31 (β- and η-equality). We say that a λ-theory L satisfies β-equality (or that it is a λ-theory with β) if $\rho_n \circ \lambda_n = \mathrm{id}_{L_n}$ for all n. We say that is satisfies η-equality if $\lambda_n \circ \rho_n = \mathrm{id}_{L_{n+1}}$ for all n.

DEFINITION 32 (λ-theory morphism). A *morphism* F between λ-theories L and L' is an algebraic theory morphism F such that

$$F_n(\lambda_n(f)) = \lambda_n(F_{n+1}(f))$$
$$\rho_n(F_n(g)) = F_{n+1}(\rho_n(g))$$

for all $f : L_{n+1}$ and $g : L_n$.

REMARK 16. The category of lambda theories has all limits, with the underlying algebraic theory of a limit being the limit of the underlying algebraic theories.

DEFINITION 33. A λ-theory algebra or presheaf is an algebra or presheaf for the underlying algebraic theory.

5. Examples

There is a lot of different examples of algebraic theories and their algebras. Some of these even turn out to be λ-theories. In this section, we will discuss a couple of these.

5.1. The free algebraic theory on a set.

EXAMPLE 2. Let S be a set. We can construct an algebraic theory $F(S)$ by taking $F(S)_n = S \sqcup \{1, \dots, n\}$ with projections $x_i = i$ and substitution

$$i \bullet g = g_i \qquad\qquad s \bullet g = s$$

for $i : \{1, \dots, n\}$ and $s : S$.

If we have a function $f : S \to S'$, we get a morphism $F(f) : F(S) \to F(S')$ given by

$$F(f)_n(i) = i \qquad\qquad F(f)_n(s) = f(s)$$

for $i : \{1, \dots, n\}$ and $s : S$.

Also, F obviously respects the identity and substitution morphisms, so it is a functor.

Note that we have a forgetful functor $(\cdot)_0$ that sends a morphism of algebraic theories $g : T \to T'$ to the function $f_0 : T_0 \to T'_0$.

LEMMA 22. *The algebraic theory $F(S)$ defined above, is the free algebraic theory on the set S.*

PROOF. Let T be an algebraic theory. We have an equivalence

$$\mathbf{AlgTh}(F(S), T) \cong \mathbf{Set}(S, T_0),$$

sending $f : \mathbf{AlgTh}(F(S), T)$ to $f_0 : S = S \sqcup \emptyset \to T_0$ (this is trivially natural in S and T) and $f : \mathbf{Set}(S, T_0)$ to the functions $g_n : F(S)_n \to T_n$ given by

$$g_n(i) = x_i \qquad\qquad g_n(s) = f(s) \bullet ().$$

□

The proofs that $F(S)$ is an algebraic theory and that $F(f)$ and g are algebraic theory morphisms is an easy exercise in case distinction.

COROLLARY 4. *$F(\emptyset)$ is the initial algebraic theory.*

PROOF. For $S = \emptyset$, the equivalence of hom-sets becomes

$$\mathbf{AlgTh}(F(\emptyset), T) \cong \mathbf{Set}(\emptyset, T_0)$$

and the latter has exactly one element. □

LEMMA 23. *There is an adjoint equivalence between the category $\mathbf{Alg}_{F(S)}$ and the coslice category $S \downarrow \mathbf{Set}$.*

PROOF. For the equivalence, we send a $F(S)$-algebra A to the set A with morphism $s \mapsto s \bullet ()$. An algebra morphism $f : A \to B$ is sent to the coslice morphism $f : (S \to A) \to (S \to B)$. This constitutes a functor.

Note that the category of $F(S)$-algebras is univalent.

Also, the functor is fully faithful, since one can show that for $F(S)$-algebras, the coslice morphism $\varphi : (f : S \to A) \to (f' : S \to B)$ also has the structure of an algebra morphism $\varphi : A \to B$.

Lastly, the functor is essentially surjective, since we can lift a coslice $f : S \to X$ to a $F(S)$-algebra X, with action

$$i \bullet x = x_i \quad \text{and} \quad s \bullet x = f(s).$$

Therefore, the functor $\mathbf{Alg}_{F(S)} \to S \downarrow \mathbf{Set}$ is an adjoint equivalence.

The proofs of these facts work by simple case distinction, and by using the properties of the coslice and algebra morphisms. □

$F(\emptyset)$ is, in some sense, the smallest nontrivial algebraic theory. Then $F(S)$ is the smallest nontrivial algebraic theory that has the elements of S as constants.

5.2. The free λ-theory on a set. Like with the free algebraic theory, we will construct the free λ-theory as the smallest nontrivial λ-theory (which is the λ-calculus) with some additional constants.

Let S be a set. Consider the sequence of inductive types $(\Lambda(S)_n)_n$ with the following constructors:

$$\begin{aligned}
&\mathtt{Var}_n : \{1, \dots, n\} \to \Lambda(S)_n;\\
&\mathtt{App}_n : \Lambda(S)_n \to \Lambda(S)_n \to \Lambda(S)_n;\\
&\mathtt{Abs}_n : \Lambda(S)_{n+1} \to \Lambda(S)_n;\\
&\mathtt{Con}_n : S \to \Lambda(S)_n.
\end{aligned}$$

Define a substitution operator $\bullet : \Lambda(S)_m \times \Lambda(S)_n^m \to \Lambda(S)_n$ by induction on the first argument:

$$\begin{aligned}
\mathtt{Var}_m(i) \bullet g &= g_i;\\
\mathtt{App}_m(a, b) \bullet g &= \mathtt{App}_n(a \bullet g, b \bullet g);\\
\mathtt{Abs}_m(a) \bullet g &= \mathtt{Abs}_n(a \bullet ((g_i \bullet (x_{n+1,j})_j)_i + (x_{n+1})));\\
\mathtt{Con}_m(s) \bullet g &= \mathtt{Con}_n(s).
\end{aligned}$$

And then quotient $\Lambda(S)$ by the relation generated by

$$\mathtt{App}_m(\mathtt{Abs}_m(f), g) \sim f \bullet ((x_{n,i})_i + (g))$$

for all $f : \Lambda(S)_{n+1}$ and $g : \Lambda(S)_n$.

Example 3. We can give the sequence of sets $\Lambda(S)$ an algebraic theory structure with variables $x_{m,i} = \mathtt{Var}_m(i)$ and the substitution operator $\bullet$ defined above. We can give $\Lambda(S)$ a λ-theory structure with β-equality by taking

$$\lambda_n(f) = \mathtt{Abs}_n(f) \quad \text{and} \quad \rho_n(f) = \mathtt{App}_{n+1}(f \bullet (\mathtt{Var}_{n+1}(i))_i, \mathtt{Var}_{n+1}(n+1)).$$

Now, given a function $S \to S'$, we define a morphism $\mathbf{LamTh}(\Lambda(S), \Lambda(S'))$ by induction, sending $\mathtt{Var}(i)$, $\mathtt{App}(a, b)$ and $\mathtt{Abs}(a)$ in $\Lambda(S)$ to their corresponding elements in $\Lambda(S')$ and sending $\mathtt{Con}(s)$ to $\mathtt{Con}(f(s))$.

Note that, like with the previous example, we have a forgetful functor $\dot{()}_0 : \mathbf{LamTh} \to \mathbf{Set}$.

Lemma 24. *$\Lambda(S)$ is the free λ-theory on S.*

Proof. Let L be a λ-theory. We have an equivalence

$$\mathbf{LamTh}(\Lambda(S), L) \cong \mathbf{Set}(S, L_0),$$

sending $f : \mathbf{LamTh}(\Lambda(S), L)$ to $f_0|_S : S \to L_0$ (again, trivially natural in S and L) and conversely, $g : \mathbf{Set}(S, L_0)$ to the inductively defined $f : \mathbf{LamTh}(\Lambda(S), L)$ given by

$$\begin{aligned}
f(\mathtt{Var}(i)) &= x_i;\\
f(\mathtt{App}(a, b)) &= \rho(f(a)) \bullet ((x_{n,i})_i + (f(b)));\\
f(\mathtt{Abs}(a)) &= \lambda(f(a));\\
f(\mathtt{Con}(s)) &= g(s) \bullet ().
\end{aligned}$$

□

The proofs that $\Lambda(S)$ is indeed a λ-theory and that $\Lambda(f)$ and g are λ-theory morphisms, mainly work by definition of $\bullet$, λ and ρ, by induction on the terms of $\Lambda(S)$ and by invoking the properties of the λ-theory L.

COROLLARY 5. *The 'pure' lambda calculus is the initial λ-theory.*

PROOF. If we take $S = \emptyset$, $\Lambda(\emptyset)$ is the lambda calculus, which we will call Λ. We have, like with the free algebraic theory, that $\Lambda(\emptyset)$ is the initial λ-theory. □

5.2.1. *About Λ-algebra morphisms.*

LEMMA 25. *Let A and B be Λ-algebras and let $f : \mathbf{Set}(A, B)$ be a function that preserves the application and the Λ-definable constants:*

$$f((x_1x_2) \bullet (a, b)) = (x_1x_2) \bullet (f(a), f(b)) \quad \textit{and} \quad f(s \bullet ()) = s \bullet ()$$

for all $a, b : A$ and $s : \Lambda_0$. Then f is a Λ-algebra morphism.

PROOF. Note that for $s : \Lambda_{n+1}$ and $a : A^{n+1}$,

$$(x_1x_2) \bullet (\lambda(s) \bullet (a_i)_i, a_{n+1}) = s \bullet a.$$

By induction, we can express $s \bullet a$ using a combination of $(x_1x_2) \bullet (\cdot, \cdot)$ and $\lambda^n(s)$:

$$\begin{aligned} f(s \bullet a) &= f((x_1x_2) \bullet (\dots((x_1x_2) \bullet (\lambda^n(s) \bullet (), a_1), \dots), a_n)) \\ &= (x_1x_2) \bullet (\dots((x_1x_2) \bullet (f(\lambda^n(s) \bullet ()), f(a_1)), \dots), f(a_n)) \\ &= s \bullet (f(a_i))_i, \end{aligned}$$

so f is a Λ-algebra morphism. □

5.3. The free object algebraic theory.

EXAMPLE 4. Take a category C, with a forgetful functor $G : C \to \mathbf{Set}$ and a free functor $F : \mathbf{Set} \to C$. Let $\eta : \mathrm{id}_{\mathbf{Set}} \Rightarrow F \bullet G$ be the unit of the adjunction and let $\varphi : C(F(c), d) \cong \mathbf{Set}(c, G(d))$ be the natural equivalence of homsets.

We define an algebraic theory T with $T_n = G(F(\{1, \dots, n\}))$, projections $x_{n,i} = \eta_{\{1,\dots,n\}}(i)$. For the substitution, note that we take $t_1, \dots, t_m : T_n$, so we have $t : \{1, \dots, m\} \to G(F(\{1, \dots, n\}))$. We then take

$$s \bullet t = G(\varphi^{-1}(t))(s).$$

Now, given an object $c : C$, we can create a T-algebra $\alpha(c)$, with set $G(c)$ and action

$$s \bullet t = G(\varphi^{-1}(t))(s).$$

Also, given a morphism $f : C(c, d)$. This gives a morphism $G(f) : \alpha(c) \to \alpha(d)$. Therefore, $\alpha : C \to \mathbf{Alg}_T$ is a functor.

The proofs that T is an algebraic theory, that $G(c)$ is an algebra and that $G(f)$ is an algebra morphism mainly rely on the fact that φ is natural.

So we have a functor from C to the category of T-algebras. One can wonder whether there also is a functor the other way, or whether α is even an equivalence. This is hard to characterize precisely, but in algebra, there is a broad class of examples where the functor is an equivalence, so where C is equivalent to $\mathbf{Alg}_T$. That is probably why T is called an *algebraic theory*.

The idea is that if an object of C is a set, together with some operations between its elements, one can carefully choose some elements of T_0, T_1, T_2 etc., which act on an algebra like the specified operations.

EXAMPLE 5. For C the category of monoids, $\alpha : C \to \mathbf{Alg}_T$ is an adjoint equivalence.

Note that T_n is the free monoid on n elements. Its elements can be viewed as strings $(x_1x_5x_3x_{18} \dots x_7)$ with the characters $x_1, \dots, x_n$, with the x_i the generators of the monoid, acting as the projections of the algebraic theory.

Let A be a T-algebra. We can give A a monoid structure by taking, for $a, b : A$,

$$ab = (x_1x_2) \bullet (a, b)$$

and unit element

$$1 = () \bullet ().$$

Then the laws like associativity follow from those laws on the monoid and from the fact that the action on the algebra commutes with the substitution:

$$a(bc) = (x_1(x_2x_3)) \bullet (a, b, c) = ((x_1x_2)x_3) \bullet (a, b, c) = (ab)c.$$

Note that if we take a monoid, turn it into a T-algebra and then into a monoid again, we still have the same underlying set, and it turns out that the monoid operation and unit element are equal to the original monoid operation and unit element. Therefore, α is essentially surjective. It is also fully faithful, since any T-algebra morphism respects the action of T, which makes it into a monoid morphism. Therefore, α is an adjoint equivalence.

REMARK 17. In the same way, one can characterize groups, rings and R-algebras (for R a ring) as algebras of some algebraic theory. On the other hand, one can not use this method to describe fields as algebras for some theory T, because one would need to describe the inverse $z \mapsto z^{-1}$ operation as $t \bullet (z)$ for some $t : T_1$, with $zz^{-1} = 1$, but since the elements of the algebraic theory act on all (combinations of) elements of the algebra, one would be able to take the inverse $0^{-1} = t \bullet (0)$ with $00^{-1} = 1$, which would make no sense.

REMARK 18. Another counterexample is the category **Top** of topological spaces. We have a forgetful functor $G : \mathbf{Top} \to \mathbf{Set}$ that just forgets the topology. On the other hand, we have a free functor $F : \mathbf{Set} \to \mathbf{Top}$ which endows a set with the discrete topology. The construction above yields the inital algebraic theory $T_n = \{1, \ldots, n\}$, with an algebra action on every topological space $i \bullet (a_1, \ldots, a_n) = a_i$. Now, note that we can endow the set $\{\top, \bot\}$ with four different, nonisomorphic topologies, which all yield the same T-algebra. In other words: the T-algebra structure does not preserve the topological information. Therefore, the functor $\alpha : \mathbf{Top} \to \mathbf{Alg}_T$ is not an equivalence.

5.4. The terminal theory.

EXAMPLE 6. We can create a (somewhat trivial) algebraic theory T by taking $T_n = \{\star\}$, with projections $x_i = \star$ and substitution $\star \bullet \star = \star$. Taking $\lambda(\star) = \star$ and $\rho(\star) = \star$, we give it a λ-theory structure (with β and η-equality). Checking that this is indeed an algebraic theory and even a λ-theory is trivial.

Now, given any other algebraic theory T', there exists a unique function $T'_n \to T_n$ for every n, sending everything to $\star$. These functions actually constitute an algebraic theory morphism $T' \to T$. If T' is a λ-theory, the algebraic theory morphism is actually a λ-theory morphism. Again, checking this is trivial.

Therefore, T is the terminal algebraic theory and λ-theory.

LEMMA 26. *$\{\star\}$ is the only algebra of the terminal theory.*

PROOF. Let A be a T-algebra. First of all, we have an element $\star_A = \star_T \bullet_0 ()$. Secondly, for all elements $\star, \star' : A$, we have

$$\star = x_1 \bullet (\star, \star') = \star \bullet (\star, \star') = x_2 \bullet (\star, \star') = \star'.$$

Therefore, $A = \{\star\}$, which allows exactly one possible T-action:

$$\star \bullet (\star, \ldots, \star) = \star.$$

□

5.5. The endomorphism theory.

DEFINITION 34. Suppose that we have a category C and an object $X : C$, such that all powers X^n of X are also in C. The *endomorphism theory* $E(X)$ of X is the algebraic theory given by $E(X)_n = C(X^n, X)$ with projections as variables $x_{n,i} : X^n \to X$ and a substitution that sends $f : X^m \to X$ and $g_1, \ldots, g_m : X^n \to X$ to $f \circ \langle g_i \rangle_i : X^n \to X^m \to X$.

DEFINITION 35. Now, suppose that the exponential object X^X exists, and that we have morphisms back and forth $abs : X^X \to X$ and $app : X \to X^X$. Let, for $Y : C$, φ_Y be the isomorphism $C(X \times Y, X) \xrightarrow{\sim} C(Y, X^X)$. We can give $E(X)$ a λ-theory structure by setting, for $f : E(X)_{n+1}$ and $g : E(X)_n$,

$$\lambda(f) = abs \circ \varphi_{X^n}(f) \qquad \rho(g) = \varphi^{-1}_{X^n}(app \circ g).$$

The proofs that $E(X)$ is an algebraic theory and a λ-theory, use properties of the product, and naturality of the isomorphism φ_Y.

5.6. The theory algebra.

EXAMPLE 7. Let T be an algebraic theory and n a natural number. We can endow the T_n with a T-algebra structure, by taking the substitution operator of T as the T-action. Since this commutes with the substitution operator and the projections, T_n is a T-algebra.

5.7. The theory presheaf.

EXAMPLE 8. Let T be an algebraic theory. We can endow T with a T-presheaf structure, by taking the substitution operator of T as the action on T. Since this commutes with the substitution operator and the projections, T is a T-presheaf.

LEMMA 27. *Given an algebraic theory T and a T-presheaf Q, we have for all n a bijection of sets*

$$\varphi : \mathbf{Pshf}_T(T^n, Q) \cong Q_n.$$

PROOF. For $f : \mathbf{Pshf}_T(T^n, Q)$, take $\varphi(f) = f_n(x_1, \ldots, x_n)$.
Conversely, for all $q : Q_n$ and all $t_1, \ldots, t_n : T^n_m$ take

$$\varphi^{-1}(q)_m(t_1, \ldots, t_n) = q \bullet t.$$

□

5.8. The "+l" presheaf.

EXAMPLE 9 (The '+l' presheaf). Given a T-presheaf Q, we can construct a presheaf $A(Q, l)$ with $A(Q, l)_n = Q_{n+l}$ and, for $q : A(Q, l)_m$ and $f : T^m_n$, action

$$q \bullet_{A(Q,l)} f = q \bullet_Q ((\iota_{n,l}(f_i))_i + (x_{n+i})_i).$$

LEMMA 28. *For all l and T-presheaves Q, $A(Q, l)$ is the exponential object Q^{T^l}.*

PROOF. We will show that $A(-, l)$ constitutes a right adjoint to the functor $- \times T^l$. We will do this using universal arrows.

For Q a T-presheaf, take the arrow $\varphi : A(Q, l) \times T^l \to Q$ given by $\varphi(q, t) = q \bullet_Q ((x_{n,i})_i + t)$ for $q : A(Q, l)_n = Q_{n+l}$ and $t : T^l_n$.

Now, given a T-presheaf Q' and a morphism $\psi : Q' \times T^l \to Q$. Define $\tilde{\psi} : Q'_n \to A(Q, l)_n$ by $\tilde{\psi}(q) = \psi(\iota_{n,l}(q), (x_{n+i})_i)$.

Then ψ factors as $\varphi \circ (\tilde{\psi} \times \mathrm{id}_{T^l})$. Also, some equational reasoning shows that $\tilde{\psi}$ is unique, which proves that φ indeed is a universal arrow. □

CHAPTER 4

Previous work in categorical semantics

1. The correspondence between categories and typed λ-calculi

In [**SH80**], page 413, Scott and Lambek argue that there is a correspondence between simply typed λ-calculi and cartesian closed categories (categories with products and 'function objects').

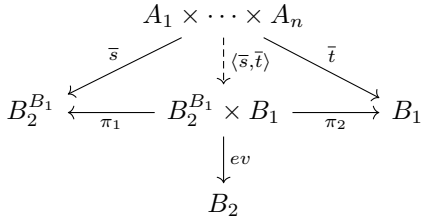
Types in the λ-calculus correspond to objects in the category.

Types $A \to B$ in the λ-calculus correspond to exponential objects B^A in the category.

Terms in the λ-calculus of type B, with free variables $x_1 : A_1, \ldots, x_n : A_n$, correspond to morphisms $A_1 \times \cdots \times A_n \to B$.

A free variable $x_i : A_i$ in a context with free variables $x_1 : A_1, \ldots, x_n : A_n$ corresponds to the projection morphism $\pi_i : A_1 \times \cdots \times A_n \to A_i$.

Given a term $s : B_1 \to B_2$ and a term $t : B_1$, both with free variables $x_1 : A_1, \ldots, x_n : A_n$, corresponding to morphisms $\overline{s} : A_1 \times \cdots \times A_n \to B_2$ and $\overline{t} : A_1 \times \cdots \times A_n \to B_1$, the application $st : B_2$ corresponds to the composite of the product morphism with the evaluation morphism $A_1 \times \cdots \times A_n \to B_2^{B_1} \times B_1 \to B_2$.



Given a term $t : B$ with free variables $x_1 : A_1, \ldots, x_n : A_n$, the abstraction $(\lambda x_n, t) : A_n \to B$ corresponds to using the adjunction corresponding to the exponential object of A_n:

$$C(A_1 \times \cdots \times A_{n-1} \times A_n, B) \simeq C(A_1 \times \cdots \times A_{n-1}, B^{A_n}).$$

2. The category of retracts

The next sections make extensive use of a category called **R**, which Hyland calls the 'category of retracts'. In this section, we will define the category, and show some properties about it.

Let L be a λ-theory. First of all, for $a_1, a_2 : L_0$, we define

$$\begin{aligned} a_1 \circ a_2 &= \lambda x_1, a_1(a_2 x_1); \\ (a_1, a_2) &= \lambda x_1, x_1 a_1 a_2; \\ \langle a_1, a_2 \rangle &= \lambda x_1, (a_1 x_1, a_2 x_1); \\ \pi_i &= \lambda x_1, x_1(\lambda x_2 x_3, x_{i+1}). \end{aligned}$$

Although, actually, since every one of these starts with a λ-abstraction, we need to lift the constants a_i to $\iota_{0,1}(a_i) : L_1$ to make the definitions above typecheck.

Note that $\pi_i(a_1, a_2) = a_i$ and $\pi_i \circ \langle a_1, a_2 \rangle = \lambda x_1, a_i x_1$, which is exactly what we would expect of a projection.

Also, note that by replacing the x_i by x_{n+i} and the $\iota_{0,1}(a_i)$ by $\iota_{n,1}(a_i)$, we obtain definitions not only for elements of L_0, but for all L_n.

DEFINITION 36. We define the category $\mathbf{R}$ as the Karoubi envelope of the monoid of the λ-terms without free variables under composition. That is, the category of idempotent functions:

$$\mathbf{R}_0 = \{A : L_0 \mid A \circ A = A\} \quad \text{and} \quad \mathbf{R}(A, B) = \{f : L_0 \mid B \circ f \circ A = f\},$$

with $\mathrm{id}_A = A$ and composition given by $\circ$.

Now, to give a bit more intuition for the objects of $\mathbf{R}$, we can pretend that an object $A : \mathbf{R}$ consists of the set of elements that satisfy $Aa = a$. Then a morphism $f : \mathbf{R}(A, B)$ gives $B(fa) = (B \circ f)a = fa$. This actually constitutes a functor from R to $\mathbf{Pshf}_L$:

DEFINITION 37. We define a functor $\varphi : \mathbf{R} \to \mathbf{Pshf}_L$ by taking

$$\varphi(A)_n = \{a : L_n \mid \iota_{0,n}(A)a = a\} \quad \text{and} \quad \varphi(f)_n(a) = \iota_{0,n}(f)a$$

for $A, B : \mathbf{R}$, $f : \mathbf{R}(A, B)$ and $a : A$. The presheaf action on $\varphi(A)$ is given by the substitution of L:

$$(a, f) \mapsto a \bullet f$$

for $f : L_n^m$ and $a : L_m$ such that $\iota_{0,m}(A)a = a$.

It turns out that this embeds $\mathbf{R}$ as a full subcategory of $\mathbf{Pshf}_L$:

LEMMA 29. *This functor φ is fully faithful.*

PROOF. Take $A, B : \mathbf{R}$. We need to show that $f \mapsto \varphi(f)$ is an equivalence between $\mathbf{R}(A, B)$ and $\mathbf{Pshf}_L(\varphi(A), \varphi(B))$. We have a function

$$\psi : \mathbf{Pshf}_L(\varphi(A), \varphi(B)) \to \mathbf{R}(A, B), \quad g \mapsto (\lambda x_1, g_1(\iota_1(A)x_1))$$

which gives us the inverse. To see that this even typechecks, note that we have

$$\iota_1(A)x_1 : \varphi(A)_1 \quad \text{and} \quad g_1(\iota_1(A)x_1) : \varphi(B)_1 \subseteq L_1.$$

Using the fact that g is a presheaf morphism, we can show that

$$B \circ \psi(g) \circ A = \psi(g),$$

and that φ and ψ are inverses. □

REMARK 19. Note that if L is a nontrivial λ-theory, $\mathbf{R}$ is not a univalent category. To see this, note, for example, that we have an object $X := \langle \pi_1, \pi_2 \rangle : \mathbf{R}$ (corresponding to the type of 'pairs' of λ-terms). Since L_0 is a set, $X = X$ is a proposition. However, X has (at least) two automorphisms:

$$\langle \pi_1, \pi_2 \rangle \quad \text{and} \quad \langle \pi_2, \pi_1 \rangle.$$

These are the identity, and the automorphism (of order 2) that swaps the elements of the pair. To see that these are indeed different morphisms, note that applying them (or their lifted versions) to $(x_{2,1}, x_{2,2})$ gives respectively $x_{2,1}$ and $x_{2,2}$, which are distinct elements by Lemma 20.

3. Scott's Representation Theorem

The correspondence in Section 4.1 between simply-typed λ-calculi and cartesian closed categories raises a question whether such a correspondence also exists for untyped λ-calculi. Definition 34 shows that in fairly general circumstances we can take one object c in a category C and consider the morphisms $t : C(c^n, c)$ as terms in an untyped λ-calculus. Hyland calls this the 'endomorphism theory' of c.

REMARK 20. To construct a *simply typed* λ-calculus from a category, we just need a cartesian closed category. In a simply typed λ-calculus, there is a lot of restriction on which terms we can apply to each other. A term of type $A \to B$ can only be applied to a term of type A, which gives a term of type B. In particular, a term can be applied only finitely many times to other terms, and every time, the result has a different type.

On the other hand, for an *untyped* λ-calculus, we need a cartesian closed category with a 'reflexive object'. This is because in the untyped λ-calculus, we can apply arbitrary terms to each other. For example, we can apply the term $(\lambda x_1, x_1 x_1)$ to itself, which would not be typable in the simply typed λ-calculus. Suppose that we have a category C and an object U such that the morphisms $C(U^n, U)$ give the untype λ-terms in n free variables, for all n. Now, given two terms $f, g : C(U^n, U)$, for the application fg, we need to consider f as a morphism in $C(U^n, U^U)$. We can do this by postcomposing with a morphism $\varphi : C(U, U^U)$. On the other hand, if $n > 0$, then $(\lambda x_n, f)$ is a morphism in $C(U^{n-1}, U^U)$, but it is a term in $n-1$ free variables, so it should be in $C(U^{n-1}, U)$. For this, we postcompose with a morphism $\psi : C(U^U, U)$. Now, for our untyped λ-calculus to have β-equality, we need $\psi \cdot \varphi = \mathrm{id}_{U^U}$, which means that the exponential U^U of U is a retract of U. This is exactly what it means for U to be a *reflexive object*: an object U in a cartesian closed category, that has a retraction onto its 'function space' U^U. Note that if we want our λ-theory to also have η-equality, the retraction must be an isomorphism.

Note that **Set** is a cartesian closed category, but that for sets X and Y, the function space X^Y has cardinality $|X|^{|Y|}$, and therefore U^U cannot be a retract of U, unless $U = \{\star\}$, in which case we have a very trivial λ-calculus: $\mathbf{Set}(U^n, U) = \{\star\}$.

During the 1960s, computer scientists sought for nontrivial examples of reflexive objects, there is a quote by Dana Scott that "Lambda-calculus has no mathematical models!"[**Sco16**]. However, in 1969, the same Dana Scott discovered that the category of (continuous) lattices with (Scott-)continuous functions between them has a nontrivial reflexive object, with the retraction onto the function spaces being even an isomorphism. Such a reflexive object D_∞ is obtained by starting with an arbitrary lattice D_0, iteratively taking $D_{n+1} = D_n^{D_n}$, with a retraction (in the 'wrong' direction) $r : D_n^{D_n} \to D_n$, and then passing to the limit $D_\infty = \lim_{\leftarrow} D_n$ (for the main result, see Theorem 4.4 in [**Sco72**], page 127).

Since Lambek showed an equivalence between simply typed λ-calculi and cartesian closed categories, and since we have a construction for an untyped λ-calculus from a cartesian closed category with a reflexive object, we can wonder whether this construction constitutes contitutes an equivalence between untyped λ-calculi and some class of categories. This question finds a partial answer in the following theorem, originally proven in a very syntactical way by Dana Scott (see [**SH80**], page 418).

THEOREM 1. *We can obtain every untyped λ-calculus as the endomorphism theory of some object in some category.*

PROOF. Let L be a λ-theory. Scott considers the category $\mathbf{R}$.

This category has a terminal object $I = \lambda x_1 x_2, x_2 : \mathbf{R}$.

The category has binary products with projections and product morphisms

$$A_1 \times A_2 = \langle p_1, p_2 \rangle, \quad p_i = A_i \circ \pi_i \quad \text{and} \quad \langle f, g \rangle.$$

The category also has exponential objects

$$C^B = \lambda x_1, B \circ x_1 \circ C$$

with evaluation morphism $\epsilon_{BC} : C^B \times B \to C$ given by

$$\epsilon_{BC} = \lambda x_1, C(\pi_1 x_1 (B(\pi_2 x_1))),$$

which is universal because we can lift a morphism $f : \mathbf{R}(A \times B, C)$ to a morphism $\varphi(f) : \mathbf{R}(A, C^B)$ given by

$$\varphi(f) = \lambda x_1 x_2, f(x_1, x_2).$$

Note that for $g : \mathbf{R}(A, C^B)$, the inverse $\varphi^{-1}(g)$ is given by

$$\epsilon \circ \langle g \circ \pi_1, B \circ \pi_2 \rangle = \lambda x_1, g(\pi_1 x_1)(\pi_2 x_1) : \mathbf{R}(A \times B, C).$$

Now, consider the object $U = \lambda x_1, x_1 : \mathbf{R}$. Note that for all $A : \mathbf{R}$, we have morphisms $A : \mathbf{R}(U, A)$ and $A : \mathbf{R}(A, U)$, which exhibit A as a retract of U. In particular, U^U is a retract of U, so U is a reflexive object.

Therefore, $E(U)$, the endomorphism theory of U, has a λ-theory structure. Note that the finite powers of U in $\mathbf{R}$ are given by $U^0 = I$ and $U^{n+1} = U^n \times U$.

We have $E(U)_n = \mathbf{R}(U^n, U) = \{f : L_0 \mid U \circ f \circ U^n = f\}$. The variables of $E(U)$ are the projections of U^n:

$$q_{n,i} = \pi_2 \circ \underbrace{\pi_1 \circ \cdots \circ \pi_1}_{n-i}.$$

The substitution is given by composition with the product morphism:

$$f \bullet g = f \circ \langle \langle \langle I, g_1 \rangle, \dots \rangle, g_n \rangle.$$

We have $U^U = \lambda f, U \circ f \circ U = \lambda x_1 x_2, x_1 x_2$. Using the equivalence $\mathbf{R}(U^n \times U, U) \simeq \mathbf{R}(U^n, U^U)$ and the retraction $U^U : U \to U^U$, the abstraction and application λ and ρ are given by

$$\lambda(f) = \lambda x_1 x_2, \iota_{0,2}(f)(x_1, x_2), \quad \rho(g) = \lambda x_1, \iota_{0,1}(g)(\pi_1 x_1)(\pi_2 x_1).$$

for $f : \mathbf{R}(U^{n+1}, U)$ and $g : \mathbf{R}(U^n, U)$.

Now, we have bijections $\psi_0 : E(U)_0 \xrightarrow{\sim} L_0$, given by

$$\psi_0(f) = f(\lambda x_1, x_1) \quad \text{and} \quad \psi_0^{-1}(g) = \lambda x_1, \iota_{0,1}(g).$$

We can extend this to any n, by reducing any term to a constant by repeatedly using λ, then applying the bijection, and then lifting it again using ρ. Explicitly, we obtain

$$\psi_n(f) = \iota_{0,n}(f)((((\lambda x_{n+1}, x_{n+1}), x_1), \dots), x_n), \quad \text{and} \quad \psi_n^{-1}(g) = \lambda x_1, g \bullet (q_i x_1)_i.$$

It is not hard to verify that this is indeed a bijection, using at one point the fact that $f : \mathbf{R}(U^n, U)$ is defined by $f \circ U^n = f$, for

$$U^n = \lambda x_1, (((\lambda x_2, x_2), q_{n,1} x_1), \dots, q_{n,n} x_1).$$

It is also pretty straightforward to check that

$$\psi(q_{n,i}) = x_i, \qquad \psi(f) \bullet (\psi(g_i))_i = \psi(f \bullet g),$$
$$\psi(\lambda(h)) = \lambda(\psi(h)), \qquad \psi(\rho(h')) = \rho(\psi(h'))$$

for $f : E(U)_m$, $g : E(U)_n^m$, $h : E(U)_{n+1}$ and $h' : E(U)_n$. Therefore, ψ is an isomorphism of λ-theories. □

4. The Taylor Fibration

In his dissertation, Paul Taylor shows that $\mathbf{R}$ is not only cartesian closed, but also *relatively cartesian closed*.

In Section 1.6, we studied internal and external representations of families of objects in a category and how they behaved under substitutions (pullbacks). This was to arrive at a definition for dependent products and sums, as the right and left adjoints to the pullback (or substitution) functor $\alpha^* : (C \downarrow A) \to (C \downarrow B)$ along some morphism $\alpha : C(B, A)$.

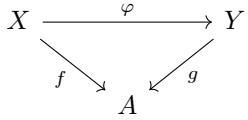
Now, some categories are not locally cartesian closed. That is: not all substitution functors α^* exist or have a right adjoint. One way to look at this, is that not all morphisms $X \to A$ represent a family of objects. In these categories, we can carefully choose a subset of the morphisms to represent our indexed families. We will call a morphism representing an indexed family a *display map*. In most cases, we have quite a bit of choice which maps we want to take as our display maps. However, to make sure that indexed families are well-behaved, a class of display maps needs to have some properties:

(1) The pullback of a display map along any morphism exists and is a display map.
(2) The composite of two display maps is a display map.
(3) C has a terminal object and any terminal projection is a display map.

REMARK 21. A 'maximal' example of a class of display maps is, for example, in the category **Set**, where we can take our class of display maps to equal the class of all morphisms of C.

REMARK 22. A 'minimal' example of a class of display maps is the class of (maps isomorphic to) product projections in a category with finite products. In this case, all indexed families are constant, and then dependent sums and products become binary products and exponential objects.

Now, let C be a category with a class of display maps D. We will denote the class of display maps from X to A with $D(X, A) \subseteq C(X, A)$. For any $A : C$, we get a category $(C \downarrow_D A)$ as a full subcategory of the slice category $(C \downarrow A)$, with as objects the display maps $f : D(X, A)$. A morphism from $f : D(X, A)$ to $g : D(Y, A)$ is a morphism $\varphi : C(X, Y)$ such that the following diagram commutes.



Note that for the terminal object $I : C$, $(C \downarrow_D I)$ is still equivalent to C, since every terminal projection is a display map.

Also note that since display maps are closed under pullbacks, the pullback functor $(C \downarrow A) \to (C \downarrow B)$ for $\alpha : C(B, A)$ restricts to a pullback functor $\alpha^* : (C \downarrow_D A) \to (C \downarrow_D B)$.

Note that since composing two display maps gives a display map again, and since the dependent sum is given by postcomposition, α^* has a left adjoint for all display maps α. That is: the fiber categories $(C \downarrow_D A)$ have dependent sums over display maps.

The question whether the fiber categories $(C \downarrow_D A)$ also have dependent products over display maps, brings us to the definition of relative cartesian closedness.

DEFINITION 38. A category C is *cartesian closed relative* to a class of display maps D, if the substitution functors α^* along display maps have right adjoints.

Now, analogously to 3, Taylor shows:

LEMMA 30. *If a category C is cartesian closed relative to a class of display maps D, then the fiber categories $(C \downarrow_D A)$ are cartesian closed and the substitution functors α^* preserve this structure.*

PROOF. Taylor proves this in a series of lemmas leading up to §4.3.7 in [**Tay86**]. It is also proved as Proposition 6 of [**HP89**]. □

(TODO) Displayed category?

4.1. Taylor's proof. First of all, Taylor considers 'functions' $X : A \to \mathbf{R}_0$ for some $A : \mathbf{R}_0$. He characterizes these functions as the λ-terms that satisfy $X \circ A = X$ and $\lambda x_1, (Xx_1) \circ (Xx_1) = X$. Taylor then constructs global dependent sums as

$$\sum_A X = \lambda x_1, (A(\pi_1 x_1), X(\pi_1 x_1)(\pi_2 x_1)).$$

Note that with the embedding in Definition 37, we can consider this as consisting of pairs (a, x) with $a : A$ and $x : Xa$, which is exactly what we would expect from a dependent sum.

Definition 39. For $\mathbf{R}$, we will take a morphism $f : X \to A$ to be a display map if we have some $Y : A \to R_0$ like above, and some isomorphism $g : X \xrightarrow{\sim} \sum_A Y$ such that $f = g \cdot p_1$ with $p_1 : \sum_A Y \to A$ given by $\lambda x_1, A(\pi_1 x_1)$.

Remark 23. Hyland actually gives a different characterization of Taylor's display maps. He claims that Taylor takes the display maps $X \to A$ to be the retracts in $(C \downarrow A)$ of $p_1 : A \times U \to A$, the projection onto the first coordinate. The two characterizations are equivalent:

Given $Y : A \to \mathbf{R}_0$, Ya is a retract of U for all a. Concretely, both morphisms $r : A \times U \to \sum_A Y$ and $s : \sum_A Y \to A \times U$ are given by $\sum_A Y$ and these commute with the projection to A. Therefore, if we have an isomorphism $g : X \xrightarrow{\sim} \sum_A Y$ in $(C \downarrow A)$, then $\sum_A Y \cdot g^{-1}$ and $g \cdot \sum_A Y$ make X into a retract of $A \times U$.

Conversely, given a retraction $r : A \times U \to X$ and section $s : X \to A \times U$ in $(C \downarrow A)$, the term

$$\lambda x_1 x_2, \pi_2(s(r(x_1, x_2)))$$

gives a function $Y : A \to \mathbf{R}_0$. Using the properties of r and s and the definition of $\sum_A Y$, one can show that r gives a morphism $\sum_A Y \to X$ and s gives a morphism $X \to \sum_A Y$, and that $r \circ s = \mathrm{id}_{\sum_A Y}$. Combined with the fact that $s \circ r = \mathrm{id}_X$, this shows that X is isomorphic to $\sum_A Y$.

Remark 24. Recall that if L is nontrivial, $\mathbf{R}$ is not univalent, and the existence of $Y : A \to \mathbf{R}_0$ with the isomorphism $g : X \xrightarrow{\sim} \sum_A Y$ is not a proposition. Take, for example, $Y : I \to \mathbf{R}_0$ given by $(\lambda x_1, U \times U)$ for I the terminal object. Recall that $(C \downarrow I)$ is equivalent to C. Under this equivalence $\sum_I Y$ is equivalent to $U \times U$. As mentioned before, $\sum_I Y$ has (at least) two distinct isomorphisms to itself: the identity and the isomorphism that swaps both sides of the product.

In a similar way, being a retract of $A \times U$ is not a proposition. Therefore, if we want the class of display maps to really be a subclass of the class of morphisms, we need the existence of $Y : A \to \mathbf{R}_0$ and the isomorphism $g : X \to \sum_A Y$, or the existence of the retraction $A \times U \to X$, to mean *mere existence* in this case.

This brings us to Taylor's theorem about $\mathbf{R}$:

Theorem 2. $\mathbf{R}$ *is cartesian closed, relative to this class of display maps.*

Proof. In the spirit of Scott, Taylor gives a very syntactical proof. □

(TODO) Relate indexed types and fibrations.

CHAPTER 5

Hyland's paper

1. Scott's Representation Theorem

THEOREM 3. *Any λ-theory L is isomorphic to the endomorphism λ-theory $E(L)$ of $L : \mathbf{Pshf}_L$, which is L, viewed as a presheaf in its own presheaf category.*

PROOF. First of all, remember that L is indeed exponentiable and that $L^L = A(L, 1)$. Now, since L is a λ-theory, we have sequences of functions back and forth $\lambda_n : A(L, 1)_n \to L_n$ and $\rho_n : L_n \to A(L, 1)_n$. These commute with the L-actions, so they constitute presheaf morphisms and $E(L)$ is indeed a λ-theory.

Lemma 27 gives a sequence of bijections $\varphi_n : \mathbf{Pshf}_L(L^n, L) \cong L_n$ for all n, sending $F : \mathbf{Pshf}_L(L^n, L)$ to $F(x_1, \ldots, x_n)$, and conversely sending $s : L_n$ to $((t_1, \ldots, t_n) \mapsto s\bullet(t_1, \ldots, t_n))$. It considers λ-terms in n variables as n-ary functions on the λ-calculus. Therefore, it should come as no surprise that φ preserves the x_i, $\bullet$, ρ and λ, which makes it into an isomorphism of λ-theories and this concludes the proof. □

2. Locally cartesian closedness of the category of retracts

DEFINITION 40 (Category of retracts). The category of retracts for a λ-theory L is the category with objects $f : L_n$ such that $f \bullet f = f$ and it has as morphisms $g : f \to f'$ the terms $g : L_n$ such that $f' \bullet g \bullet f = g$. The object $f : L_n$ has identity element f, and we have composition $g \circ g' = g \bullet g'$. These are morphisms **(TODO)**

LEMMA 31. *The category of retracts is indeed a category.*

PROOF. **(TODO)** □

THEOREM 4. *The category of retracts is locally cartesian closed **(TODO)** .*

3. Equivalences

4. Terms of a Λ-algebra

Let A be an algebra for the initial λ-theory Λ. We will assume that Λ (and therefore, any λ-theory) satisfies β-equality.

The Λ-algebra structure gives the terms of A quite a lot of behaviour. For example, we can define 'function application' as

$$ab = (x_1 x_2) \bullet (a, b)$$

and composition as

$$a \circ b = (x_1 \circ x_2) \bullet (a, b)$$

for $a, b : A$, with $x_1 \circ x_2 = \lambda x_3, x_1(x_2 x_3) : \Lambda_2$.

REMARK 25. Recall that in Example 5, we constructed an algebraic theory T with a monoid structure. This allowed us to define a monoid operation on T-algebras as well. We then were able to transfer associativity of the operation on the T_n to associativity of the operation on the algebras. In exactly the same way, the function composition on A is associative because composition on Λ_n is associative.

DEFINITION 41. We can consider the sets of elements of A that behave like functions in n variables:

$$A_n = \{a : A \mid (\lambda x_2 x_3 \dots x_{n+1}, x_1 x_2 x_3 \dots x_{n+1}) \bullet a = a\}.$$

DEFINITION 42. Take $\mathbf{1}_n = (\lambda x_1 \dots x_n, x_1 \dots x_n) \bullet () : A$.

REMARK 26. Some straightforward rewriting, shows that for all $a : A$,

$$\mathbf{1}_n \circ a = (\lambda x_2 x_3 \dots x_{n+1}, x_1 x_2 \dots x_{n+1}) \bullet a.$$

In other words, $A_n = \{a : A \mid \mathbf{1}_n \circ a = a\}$.

REMARK 27. Also note that $\mathbf{1}_n \circ a \circ \mathbf{1}_n = \mathbf{1}_n \circ a$, so for $a : A_n$, $a \circ \mathbf{1}_n = a$.

LEMMA 32. *For $t : \Lambda_{m+n}$ and $a_1, \dots, a_m : A$, we have $(\lambda^n t) \bullet (a_1, \dots, a_m) : A$ and we have*

$$\mathbf{1}_n \circ ((\lambda^n t) \bullet (a_1, \dots, a_m)) = (\lambda^n t) \bullet (a_1, \dots, a_m),$$

so $(\lambda^n t) \bullet (a_1, \dots, a_m) : A_n$.

PROOF. This follows by straightforward rewriting. □

COROLLARY 6. *By the previous remark,*

$$((\lambda^n t) \bullet (a_1, \dots, a_m)) \circ \mathbf{1}_n = (\lambda^n t) \bullet (a_1, \dots, a_m).$$

COROLLARY 7. *In particular, $\mathbf{1}_m \circ \mathbf{1}_n = \mathbf{1}_{\max(m,n)}$. From this, it follows that $A_m \subseteq A_n$ for $m \leq n$. It also follows that $a \mapsto \mathbf{1}_n \circ a$ gives a function from A to A_n (and also from $A_m \subseteq A$ to A_n).*

5. The Fundamental Theorem of the λ-calculus

The fundamental theorem states that there is an equivalence of categories

$$\mathbf{LamTh} \cong \mathbf{Alg}_\Lambda.$$

We will prove this by showing that there is a fully faithful and essentially surjective functor.

5.1. The functor.

DEFINITION 43. For all n, we have a functor from lambda theories to Λ-algebras. It sends the λ-theory L to the L-algebra L_n and then turns this into a Λ-algebra via the morpism $\Lambda \to L$. It sends morphisms $f : L \to L'$ to the algebra morphism $f_n : L_n \to L'_n$.

5.2. Lifting Λ-algebras.

DEFINITION 44 (The monoid of a Λ-algebra). Now we make A_1 into a monoid under composition $\circ$ with unit $\mathbf{1}_1$. The fact that this is a monoid follows from the remarks in the last section.

Recall that we have an equivalence $[C_{A_1}^{\mathrm{op}}, \mathbf{Set}] \cong \mathbf{RAct}_{A_1}$.

DEFINITION 45. Now, composition $\circ$ gives a right A_1-action on the A_n, so we have $A_n : \mathbf{RAct}_{A_1}$.

LEMMA 33. *We have $U_{A_1}^{U_{A_1}} \cong A_2$ in $\mathbf{RAct}_{A_1}$.*

PROOF. Recall that $U_{A_1}^{U_{A_1}}$ consists of the set of A_1-equivariant morphisms $U_{A_1} \times U_{A_1} \to U_{A_1}$.

We have a bijection $\varphi : A_2 \xrightarrow{\sim} U_{A_1}^{U_{A_1}}$, given by

$$\varphi(a)(b, b') = (\lambda x_4, x_1(x_2x_4)(x_3x_4)) \bullet (a, b, b'),$$

(TODO) : relate to product in category of retracts.

with an inverse given by

$$\varphi^{-1}(f) = \lambda x_1 x_2, f(p_1, p_2)(\lambda x_3, x_3 x_1 x_2)$$

(TODO) : Make more explicitly that we use $\bullet$ here. for $p_i = \lambda x_1, x_1(\lambda x_2 x_3, x_{i+1})$. Note that for terms c_1, c_2, we have $p_i(\lambda x_1, x_1 c_1 c_2) = c_i$.

This is an inverse, because given $f : U_{A_1}^{U_{A_1}}$ and $(a_1, a_2) : U_{A_1} \times U_{A_1}$, we have

$$\varphi(\psi(f))(a_1, a_2) = f(p_1, p_2) \circ q = f(p_1 \circ q, p_2 \circ q) = f(a_1, a_2)$$

for $q = \lambda x_1, (\lambda x_2, x_2(a_1 x_1)(a_2 x_1))$. In the last step of this proof, we use, among other things, the fact that the $a_i : A_1$ and therefore $\lambda x_1, a_i x_1 = a_i$.

Some straightforward rewriting shows that for $a : A_2$, we have $\psi(\varphi(a)) = a$. In the last step of this proof, we use the fact that $a : A_2$ and therefore $\lambda x_1 x_2, a x_1 x_2 = a$.

Therefore, φ is a bijection and, as it turns out, an isomorphism. □

DEFINITION 46 (Construction of the λ-theory). Since $\mathbf{RAct}_{A_1}$ has products, the algebraic theory $E(U_{A_1})$ exists.

Recall that $A_2 \subseteq A_1$ and that $a \mapsto \mathbf{1}_2 \circ a$ gives a function from A_1 to A_2, which is, by definition, the identity on A_2. This gives $E(U_{A_1})$ a λ-theory structure with β-equality.

5.3. Lifting algebra morphisms.

DEFINITION 47 (Pullback functor on presheaves for a Λ-algebra). A Λ-algebra morphism preserves $\mathbf{1}_n$ and $\circ$, so it sends elements of A_n to A'_n. In particular, it gives a monoid morphism $f : A_1 \to A'_1$.

Therefore, as described in Section 1.10, we get pullback and pushforward functors $f_* : \mathbf{RAct}_{A'_1} \to \mathbf{RAct}_{A_1}$ and $f^* : \mathbf{RAct}_{A_1} \to \mathbf{RAct}_{A'_1}$.

REMARK 28. By Lemma 16, we have $f^*(U_{A_1}) \cong U_{A'_1}$.

LEMMA 34. *f^* preserves finite products.*

PROOF. We will show that f^* preserves binary products and the terminal object.

We use Lemma 17 to show that f^* preserves the terminal object. We take

$$a_0 = (\lambda x_1 x_2, x_2) \bullet () : A_1 \quad \text{and} \quad a'_0 = (\lambda x_1 x_2, x_2) \bullet () : A'_1$$

and a'_0 is weakly terminal because for all $a : A_1$, we have $f(a_0) \circ a = a'_0$.

We use Lemma 18 to show that f^* also preserves the product. Therefore, given $a_1, a_2 : A_1$.

(**(TODO)** : relate to product)

Take $a = \lambda x_1 x_2, x_2(a_1 x_1)(a_2 x_1)$ and $\pi_i = \lambda x_1, x_1(\lambda x_2 x_3, x_{i+1})$. We have $a_i = f(\pi_i) \circ a$. Now, for some $a' : A'_1$, $\pi'_1, \pi'_2 : A_1$ such that $a_i = f(\pi'_i) \circ a'$, take $m = \lambda x_1 x_2, x_2(\pi'_1 x_1)(\pi'_2 x_1)$. Then $\pi_i \circ m = \pi'_i$ and $f(m) \circ a' = a$, so (a, π_1, π_2) is weakly terminal and f^* preserves binary products.

Since any finite product is (isomorphic to) a construction with a repeated binary product and the terminal object, the fact that f^* preserves binary products and the terminal object shows that f^* preserves all finite products. □

DEFINITION 48. Since f^* preserves finite products, given an element of $g : E(U_{A_1})_n = \mathbf{RAct}_A(U_{A_1}^n, U_{A_1})$, we get

$$f^*(g) : \mathbf{RAct}_{A'}(f(U_{A_1}^n), f(U_{A_1})) \cong \mathbf{RAct}_{A'}((U_{A_1'})^n, U_{A_1'}) = E(U_{A_1'})_n$$

so we have a morphism $f^* : \mathbf{LamTh}(E(U_{A_1}), E(U_{A_1'}))$.

REMARK 29. The fact that f^* preserves the variables and substitution is not very hard, since these are just defined in terms of finite products U_{A_1} and f^* preserves finite products and U_{A_1}.

However, showing that it is a λ-theory morphism is really hard. Hyland claims

> f preserves $\mathbf{1}_n$, which determines the function space as a retract of the universal. So f^* preserves the retract and the result follows.

This indeed covers the core of the argument, but actually verifying that it works is much more complicated: We a natural equivalence and isomorphisms

$$\begin{aligned}
\alpha_A : \mathbf{RAct}_{A_1}(X \times Y, Z) &\xrightarrow{\sim} \mathbf{RAct}_A(X, Z^Y);\\
\beta : f^*(U_{A_1}) &\xrightarrow{\sim} U_{A_1'};\\
\bar{\gamma} : f^*(A \times B) &\xrightarrow{\sim} f^*(A) \times f^*(B);\\
\gamma_n : f^*(X^n) &\xrightarrow{\sim} f^*(X)^n;\\
\delta_A : U_{A_1}^{U_{A_1}} &\xrightarrow{\sim} A_2.
\end{aligned}$$

with $\gamma_{n+1} = \bar{\gamma} \cdot (\gamma_n \times \mathrm{id}_X)$. Then, for $s : E(U_{A_1})_{n+1}$, we have $\lambda_A(s) = \alpha_A(s) \cdot \delta_A \cdot \pi_A : \mathbf{RAct}_{A_1}(U_{A_1}^n, U_{A_1})$ for the projection $\pi_A : A_2 \to U_{A_1}$. However, note that $f^*(\lambda_A(s)) : \mathbf{RAct}_{A_1'}(f^*(U_{A_1'}^n), f^*(U_{A_1'}))$. To compare this to $\lambda_{A'}(\dots)$, we actually need to take

$$(\beta^{-1})^n \cdot \gamma_n^{-1} \cdot f^*(\lambda_A(s)) \cdot \beta : \mathbf{RAct}_{A_1'}(U_{A_1}^n, U_{A_1}).$$

On the other hand, we also have $f^*(s) : \mathbf{RAct}_{A_1'}(f^*(U_{A_1}^{n+1}), f^*(U_{A_1}))$ and we cannot apply $\lambda_{A'}$ directly. Instead, the term that we end up with is

$$\lambda_{A'}((\beta^{-1})^{n+1} \cdot \gamma_{n+1}^{-1} \cdot f^*(s) \cdot \beta) : \mathbf{RAct}_{A_1'}(U_{A_1}^n, U_{A_1}).$$

So the equality that we need to prove is (using functoriality of f^* and naturality of $\alpha_{A'}$):

$$\begin{aligned}
&(\beta^{-1})^n \cdot \gamma_n^{-1} \cdot f^*(\alpha_A(s)) \cdot f^*(\delta_A \cdot \pi_A) \cdot \beta\\
&= (\beta^{-1})^n \cdot \gamma_n^{-1} \cdot \alpha_{A'}((\mathrm{id}_X^n \times \beta^{-1}) \cdot \bar{\gamma}^{-1} \cdot f^*(s)) \cdot \beta^{U_{A_1}} \cdot \delta_{A'} \cdot \pi_{A'}.
\end{aligned}$$

Because of the complicated definition of each of these terms, I was unable to verify the correctness of this statement (and the similar statement about $\rho : E(U_{A_1})_n \to E(U_{A_1})_{n+1}$) within a day, even though I am willing to believe Hyland's claim. **(TODO)**

LEMMA 35. *We have a bijection $\varphi : E(U_{A_1})_0 \cong A$, sending s to $x_1(\lambda x_2, x_2) \bullet (s(\star))$.*

PROOF. Using 12, we showed that $E(U_{A_1})_0$ corresponds to the set of A_1-equivariant elements of U_{A_1}, sending s to $s(\star)$. Now, for an A_1-equivariant element a,

$$(\lambda x_2, x_1(\lambda x_3, x_3)) \bullet a = a \circ ((\lambda x_1 x_2, x_2) \bullet ()) = a$$

so we can send a to A as $(x_1(\lambda x_2, x_2)) \bullet a$ and it has (two-sided) inverse $(\lambda x_2, x_1) \bullet a$. □

We want to show that φ is an isomorphism of Λ-algebras. We use Lemma 25 for this, so we need to show that it preserves the application and the Λ-definable constants.

LEMMA 36. *For φ as above, we have for all $a, b : E(U_{A_1})_0$, we have*

$$\varphi((x_1 x_2) \bullet (a, b)) = (x_1 x_2) \bullet (\varphi(a), \varphi(b)).$$

PROOF. For $a, b : E(U_{A_1})_0$, we have, for the isomorphism $\delta : \mathbf{RAct}_{A_1}(U_{A_1}^{U_{A_1}}, A_2)$,

$$\begin{aligned}
\varphi((x_{2,1}x_{2,2}) \bullet (a, b)) &= \varphi(\rho(x_{1,1}) \bullet (a, b)) \\
&= (x_1(\lambda x_2, x_2)) \bullet (\langle a, b\rangle \cdot ((s, t) \mapsto (\delta^{-1}(\mathbf{1}_2 \circ s))(1, t)))(\star) \\
&= (x_1(\lambda x_2, x_2)) \bullet (\lambda x_3, x_1 x_3(x_2 x_3)) \bullet (a(\star), b(\star)) \\
&= (x_1(\lambda x_3, x_3)(x_2(\lambda x_3, x_3))) \bullet (a(\star), b(\star)) \\
&= (x_1 x_2) \bullet (\varphi(a), \varphi(b))
\end{aligned}$$

and this concludes the proof. □

LEMMA 37. *For φ as above, we have for all $s : \Lambda_0$,*

$$\varphi(s \bullet ()) = s \bullet ().$$

PROOF. **(TODO)** □

LEMMA 38. *We have an isomorphism $U_{L_0} \cong L$.*

PROOF. **(TODO)** □

LEMMA 39. *This is indeed an isomorphism of λ-theories.*

PROOF. **(TODO)** □

THEOREM 5. *There exists an adjoint equivalence between the category of λ-theories, and the category of algebras of Λ.*

PROOF. We will show that the functor $L \mapsto L_0$ is an equivalence of categories.

It is essentially surjective, because L is isomorphic **(TODO)** to $E(U_{A_1})$.

Now, given morphisms $f, f' : L \to L'$. Suppose that $f_0 = f'_0$. Suppose that L and L' have β-equality. Then, given $l : L_n$, we have

$$f_n(l) = \rho^n(\lambda^n(f_n(l))) = \rho^n(f_0(\lambda^n(l))) = \rho^n(f'_0(\lambda^n(l))) = \rho^n(\lambda^n(f'_n(l))) = f'_n(l),$$

so the functor is faithful.

The functor is full because a Λ-algebra morphism $f : A \to A'$ induces a functor $f^* : \mathbf{RAct}_{A'} \to \mathbf{RAct}_A$, and via left Kan extension we get a left adjoint $f^* : \mathbf{RAct}_A \to \mathbf{RAct}_{A'}$ with $f^*(A_1) \cong A'_1$. Now, f^* preserves (finite) products, so we have maps $\mathbf{RAct}_A(A_1^n, A_1) \to \mathbf{RAct}_{A'}((A'_1)^n, A'_1)$ and so a map $E(U_{A_1}) \to E(U_{A'_1})$. This map, when restricted to a map $\mathbf{RAct}_A(1, A_1) \to \mathbf{RAct}_{A'}(1, A_1)$, and transported along the isomorphism $a \mapsto aI$ **(TODO)** , is equal to f **(TODO)** . □

6. An alternative proof for the fundamental theorem

7. Theory of extensions

LEMMA 40. *The category of T-algebras has coproducts.*

PROOF. **(TODO)** □

DEFINITION 49 (Theory of extensions). Let T be an algebraic theory and A a T-algebra. We can define an algebraic theory T_A called 'the theory of extensions of A' with $(T_A)_n = T_n + A$. The left injection of the variables $x_i : T_n$ gives the variables. Now, take $h : (T_n + A)^m$. Sending $g : T_m$ to $\varphi(g) := g \bullet h$ gives a T-algebra morphism $T_m \to T_n + A$ since

$$\varphi(f \bullet g) = f \bullet g \bullet h = f \bullet (g_i \bullet h) = f \bullet (\varphi(g_i))_i.$$

This, together with the injection morphism of A into $T_n + A$, gives us a T-algebra morphism from the coproduct: $T_m + A \to T_n + A$. We especially have a function on sets $(T_m + A) \times (T_n + A)^m \to T_n + A$, which we will define our substitution to be.

LEMMA 41. *T_A is indeed an algebraic theory.*

PROOF. **(TODO)** □

CHAPTER 6

The formalization

1. Statistics

2. Components

3. Displayed categories

3.1. Fibrations.

3.2. Defininig objects by their category instead of the other way around.

3.3. _ax for categories and their objects.

3.4. Cartesian vs cartesian' for products.

3.5. Limits.

3.5.1. *Limits in a fibered category.*

4. Inductive types

5. The formalization of the λ-calculus

Defining Lambda Calculus in a different way (not as an axiomatized HIT) - As set quotient instead of HIT - With a signature

6. Tuples

$stnm \to A$ vs vec A

7. Products

$T \times (T \times \cdots \times T)$ vs $T \times T^n vs T^(Sn)$ Terminal as product over empty set over any set with a function to empty.

8. The $n + p$-presheaf

L (S n) (for lambda) vs L (n + 1) (stemming from the naive implementation of the L (n + p) presheaf)

9. Quotients

Quotients (by hrel or eqrel) vs coproducts (generalizing to arbitrary category with coproduct) vs a category with some structure

10. The Karoubi envelope

KanExtension instead of specific construction at KaroubiEnvelope

11. Univalence

Univalence bewijzen via isweqhomot vs direct

12. Equality, Iso's and Equivalence (of categories)

It is important to choose the right kind of equality to prove.

Bibliography

[ACU14] Thosten Altenkirch, James Chapman, and Tarmo Uustalu. Monads need not be endofunctors. 2014.

[AKS15] BENEDIKT AHRENS, KRZYSZTOF KAPULKIN, and MICHAEL SHULMAN. Univalent categories and the rezk completion. *Mathematical Structures in Computer Science*, 25(5):1010–1039, January 2015.

[ARV10] J. Adámek, J. Rosický, and E. M. Vitale. *Algebraic Theories: A Categorical Introduction to General Algebra*. Cambridge Tracts in Mathematics. Cambridge University Press, 2010.

[AW23] Benedikt Ahrens and Kobe Wullaert. Category theory for programming, 2023.

[BBGS20] Marc Bezem, Ulrik Buchholtz, Daniel R. Grayson, and Michael Shulman. Construction of the circle in unimath, 2020.

[Fre72] Peter Freyd. Aspects of topoi. *Bulletin of the Australian Mathematical Society*, 7(1):1–76, August 1972.

[HP89] J. Martin E. Hyland and Andrew M. Pitts. The theory of constructions: Categorical semantics and topos-theoretic models. *Contemporary Mathematics*, 92:137–199, 1989.

[Hyl14] Martin Hyland. Towards a notion of lambda monoid. *Electronic Notes in Theoretical Computer Science*, 303:59–77, 2014. Proceedings of the Workshop on Algebra, Coalgebra and Topology (WACT 2013).

[KS06] Masaki Kashiwara and Pierre Schapira. *Categories and sheaves*, volume 332 of *Grundlehren der mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences]*. Springer-Verlag, Berlin, 2006.

[ML71] Per Martin-Löf. A theory of types. Preprint, Stockholm University, 1971.

[ML98] Saunders Mac Lane. *Categories for the working mathematician*, volume 5 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, second edition, 1998.

[nLa24] nLab authors. Grothendieck fibration. `https://ncatlab.org/nlab/show/Grothendieck+fibration`, February 2024. Revision 113.

[Sco72] Dana Scott. Continuous lattices. In F. W. Lawvere, editor, *Toposes, Algebraic Geometry and Logic*, pages 97–136, Berlin, Heidelberg, 1972. Springer Berlin Heidelberg.

[Sco16] Dana S. Scott. Greetings to the participants at "strachey 100". `https://www.cs.ox.ac.uk/strachey100/Strachey_booklet.pdf`, 2016. A talk read out at the Strachey 100 centenary conference.

[SH80] J. P. Seldin and J. R. Hindley, editors. *To H.B. Curry: Essays on Combinatory Logic and Formalism*. Academic Press, San Diego, CA, September 1980.

[Tay86] Paul Taylor. *Recursive Domains, Indexed Category Theory and Polymorphism*. PhD thesis, University of Cambridge, 1986.

[Uni13] The Univalent Foundations Program. *Homotopy Type Theory: Univalent Foundations of Mathematics*. `https://homotopytypetheory.org/book`, Institute for Advanced Study, 2013.

APPENDIX A

Alternative definitions

The literature, there are many different but equivalent definitions, carrying many different names, for the objects that called 'algebraic theories' in Section 3.1. Also, many of these different names are attached to differently defined objects in different sources. This section will showcase some of the various definitions.

In this section, we will denote the finite set $[\![n]\!] = \{1, \dots, n\}$.

1. Abstract Clone

DEFINITION 50. An algebraic theory as presented in Section 3.1, is usually called an *abstract clone*. In this thesis, outside of this specific section, we will call it algebraic theory to be consistent with the names that Hyland attaches to objects.

REMARK 30. The definition of algebraic theory that Hyland gives is closest to that of an abstract clone. However, instead of a sequence of sets $(T_n)_n$, he requires a functor $T : F \to \mathbf{Set}$ (with $F \subseteq \mathbf{FinSET}$ the skeleton category of finite sets $F_0 = \{[\![0]\!], [\![1]\!], \dots\}$), and he requires $\bullet : T_m \times T_n^m \to T_n$ to be dinatural in n and natural in m.

Using naturality, one can show that with such a functor we have $x_{n,i} = f(a)(x_{1,1})$ for the function $a(1) = i : [\![n]\!]$.

Alternatively, using the same naturality, one can show that this functor sends a morphism $a : [\![m]\!] \to [\![n]\!]$ to the function $T_m \to T_n$ given by

$$f \mapsto f \bullet (x_{n,a(i)})_i.$$

If we take this to be the definition of our functor on morphisms, the (di)naturality in m and n can be shown using the associativity and the laws about the interaction between $\bullet$ and the x_i.

Since any additional properties mean extra complexity when formalizing, and since the proofs rarely use the functor structure, we decided to reduce the functor $T : \mathbf{FinSET} \to \mathbf{Set}$ to a sequence of sets $(T_n)_n$.

2. Lawvere theory

DEFINITION 51. An *algebraic theory* as presented in [**ARV10**] is a small category with finite products.

DEFINITION 52. An *algebra* for an algebraic theory T is a finite-product-preserving functor $T \to \mathbf{Set}$.

This definition is more general than the definition of algebraic theory in Section 3.1. To make it equivalent, we have to be more specific about the objects of the category:

DEFINITION 53. A *Lawvere theory*, or *one-sorted algebraic theory* is a category L, with $L_0 = \{0, 1, \dots\}$, such that $n = 1^n$, the n-fold product.

LEMMA 42. *There is an equivalence between abstract clones and Lawvere theories.*

PROOF. Let C be an abstract clone. We construct a Lawvere theory L as follows: We have objects $L_0 = \{0, 1, \dots\}$ and morphisms $L(m,n) = C_m^n$. The identity morphism is $\mathrm{id}_n = (x_i)_i : L(n,n)$ and for $f : L(l,m)$, $g : L(m,n)$, we have composition

$$f \cdot g = (g_i \bullet f)_i : L(l,n).$$

Lastly, we have product projections $\pi_{n,i} = x_{n,i} : L(n,1)$ for all $1 \leq i \leq n$.

Conversely, if L is a Lawvere theory, we construct an abstract clone C as follows: We take $C_n = L(n,1)$. We take the $x_{n,i}$ to be the product projections $\pi_i : L(n,1)$ and we define the substitution $f \bullet g = \langle g_i \rangle_i \cdot f$ the composite of the product morphism $\langle g_i \rangle_i$ with f. □

2.1. Algebras for Lawvere Theories.

LEMMA 43. *Let C be an abstract clone, and L be its associated Lawvere theory by the equivalence given above. A C-algebra is equivalent to an algebra for L.*

PROOF. Let A be a C-algebra. We will construct a functor $F : L \to \mathbf{Set}$ as follows: We take $F(n) = A^n$. We define the action on morphisms as

$$F(f)(a) = (f_i \bullet a)_i$$

for $f : L(m,n) = L(m,1)^n$ and $a : A^m$.

Conversely, let $F : L \to \mathbf{Set}$ be a functor. We take the C-algebra A, with $A = F(1)$ and for $f : C_n = L(n,1)$ and $a : A^n$, $f \bullet a = F(f)(a)$. □

3. Cartesian Operad

DEFINITION 54. A *cartesian operad* T is a functor $T : F \to \mathbf{Set}$, together with an 'identity' element $\mathrm{id} : T(1)$ and for all $m, n_1, \dots, n_m : \mathbb{N}$, a composition map

$$T(m) \times \prod_i T(n_i) \to T\left(\sum_i n_i\right),$$

written $(f, g_1, \dots, g_m) \mapsto f[g_1, \dots, g_m]$, satisfying some identity, associativity and naturality conditions (see [**Hyl14**], Definition 2.1, for more details).

REMARK 31. One can arrive at this concept as a standalone definition, or one can view a cartesian operad as a *cartesian multicategory* with one element. A multicategory is a category in which morphisms have type $C((X_1, X_2, \dots, X_n), Y)$ instead of $C(X, Y)$. A cartesian multicategory is a multicategory in which one can permute the X_i of a morphism and has 'contraction' and 'weakening' operations:

$$C((X_1, \dots, X_i, X_i, \dots, X_n), Y) \to C((X_1, \dots, X_i, \dots, X_n), Y),$$
$$C((X_1, \dots, X_{i-1}, X_{i+1}, \dots, X_n), Y) \to C((X_1, \dots, X_{i-1}, X_i, X_{i+1}, \dots, X_n), Y).$$

LEMMA 44. *There is an equivalence between abstract clones and cartesian operads.*

PROOF. Let C be an abstract clone. We define a cartesian operad T with $T(n) = C_n$ and $T(f)(t) = t \bullet (x_{f(1)}, \dots, x_{f(m)})$ for $f : [\![m]\!] \to [\![n]\!]$ and $t : C_m$. The identity element is $x_{1,1}$ and the composition $f[g_1, \dots, g_n]$, for $f : C_m$ and $g_i : C_{n_i}$, is given by lifting all terms to $C_{\sum_i n_i}$ and then substituting:

$$f[g_1, \dots, g_n] = f \bullet (T(\iota_i)(g_i))$$

for $\iota_i : [\![n_i]\!] \hookrightarrow [\![\sum_i n_i]\!]$ the pairwise disjoint injections.

Conversely, given a cartesian operad T, we construct an abstract clone C with $C_n = T([\![n]\!])$. The variables are $x_{n,i} = \iota_{n,i}(\mathrm{id})$, for $\iota_{n,i} : [\![1]\!] \hookrightarrow [\![n]\!]$ the morphism

that sends 1 to i. The substitution $f \bullet (g_i)_i$ for $g_1, \ldots, g_m : T(n)$ is given by composing and then identifying some variables:

$$f \bullet (g_i)_i = T(\pi)(f[g_1, \ldots, g_m])$$

for $\pi : [\![mn]\!] \to [\![n]\!]$ the function that sends $i+1$ to $(i \mod n) + 1$. □

4. Relative Monad

Definition 55. Let $S : C \to D$ be a functor. A *relative monad* on S is a functor $T : C \to D$, together with a natural transformation $\eta : S \Rightarrow T$ and a 'kleisli extension' $(-)^* : D(S(X), T(Y)) \to D(T(X), T(Y))$, natural in both S and T, such that for all $f : D(S(X), T(Y))$ and $g : D(S(Y), T(Z))$,

$$\eta_X^* = \mathrm{id}_{TX}, \qquad f = \eta_X \cdot f^* \qquad \text{and} \qquad (f \cdot g^*)^* = f^* \cdot g^*.$$

Remark 32. Note that for an adjunction $F \dashv G$, $F \bullet G$ gives a monad. In the same way, there exists a notion of *relative adjunction*, from which we can obtain a relative monad (see [**ACU14**], Theorem 2.10).

Remark 33. Now, there is a result that states: There is an equivalence between abstract clones and relative monads on the embedding $\iota : \mathbf{FinSET} \hookrightarrow \mathbf{Set}$.

Note that the objects of **FinSET** are defined to be sets X, together with a proof that there exists some $n : \mathbb{N}$ and some bijection $f : X \xrightarrow{\sim} [\![n]\!]$. This existence of n and f is given by the propositional truncation $\left\| \sum_{n:\mathbb{N}}, X \xrightarrow{\sim} [\![n]\!] \right\|$.

Classically, this construction starts with "fix, for all X : **FinSET**, a bijection $f : X \xrightarrow{\sim} [\![n]\!]$". However, since X only provides *mere existence* of such a bijection without choosing one (by the propositional truncation), there is no way to obtain f without using the axiom of choice.

Now, we can partially circumvent this problem by noting that we have a fully faithful and essentially surjective embedding $F \to \mathbf{FinSET}$ (for F a skeleton of finite sets), which induces an adjoint equivalence $[F, \mathbf{Set}] \xrightarrow{\sim} [\mathbf{FinSET}, \mathbf{Set}]$, so we can lift the functor part of a relative monad on $F \to \mathbf{Set}$ to $\mathbf{FinSET} \to \mathbf{Set}$. We can also lift the natural transformation using this equivalence. However, the kleisli extension cannot be lifted using this and we are stuck.

Therefore, we will prove a modified statement:

Lemma 45. *There is an equivalence between abstract clones and relative monads on the embedding $\iota : F \hookrightarrow \mathbf{Set}$.*

Proof. Let C be an abstract clone. We define a relative monad T as follows: We take $T([\![n]\!]) = C_n$. For a morphism $a : F([\![m]\!], [\![n]\!])$. We take $T(a)(f) = f \bullet (x_{a(i)})_i$. We define $\eta_{[\![n]\!]}(i) = x_{n,i}$. Finally, for $g : \mathbf{Set}([\![m]\!], T([\![n]\!]))$, we define $g^*(f) = f \bullet g$.

Conversely, let $(T, \eta, (\cdot)^*)$ be a relative monad on the embedding $\iota : F \hookrightarrow \mathbf{Set}$. We define an abstract clone C with $C_n = T([\![n]\!])$. Substitution is defined as $f \bullet g = g^*(f)$ for $f : C_m$ and $g : C_n^m$ and we have variables $x_{n,i} = \eta_{[\![n]\!]}(i)$. □

5. Monoid in a skew-monoidal category

For details and a general treatment, see [**ACU14**], Section 3 and specifically Theorem 3.4.

Definition 56. Consider the category $[F, \mathbf{Set}]$. Note that we have a functor $\iota : F \hookrightarrow \mathbf{Set}$ and that **Set** has colimits. Therefore, we can define a 'tensor product' on functors $[F, \mathbf{Set}]$ as

$$F \otimes G = G \bullet \mathrm{Lan}_\iota F.$$

Together with the 'unit' ι, this gives $[F, \mathbf{Set}]$ a 'skew-monoidal category' structure.

DEFINITION 57. Given a (skew-)monoidal category $(C, \otimes, I)$, a *monoid* in this category is an object $T : C$, together with a 'multiplication' $\mu : C(T \otimes T, T)$ and a 'unit' morphism $\eta : C(I, T)$ satisfying a couple of laws (see [**ML98**], Section III.6).

LEMMA 46. *There exists an equivalence between relative monads on* $\iota : F \hookrightarrow$ **Set** *and monoids in the skew-monoidal category* $[F, \mathbf{Set}]$.

PROOF. A monoid in $[F, \mathbf{Set}]$ consists of an object $T : [F, \mathbf{Set}]$, together with natural transformations $\mu : T \otimes T \Rightarrow T$ and $\eta : \iota \Longrightarrow T$. We can immediately see the functor T and natural transformation η of the relative monad pop up here. The kleisli extension corresponds to μ; they are related to each other via the properties of the left Kan extension of T. □

Index