Introduction to Categorical Logic

[DRAFT: April 20, 2024]

Steve Awodey Andrej Bauer

Contents

Chapter 4

Type Theory

4.1 The Curry-Howard correspondence

Consider the following natural deduction proof in propositional calculus.

$$\dfrac{\dfrac{\dfrac{\dfrac{[(A \wedge B) \wedge (A \Rightarrow B)]^1}{A \wedge B}}{A} \qquad \dfrac{[(A \wedge B) \wedge (A \Rightarrow B)]^1}{A \Rightarrow B}}{B}}{(A \wedge B) \wedge (A \Rightarrow B) \Rightarrow B}\,{\scriptstyle (1)}$$

This deduction shows that

$$\vdash (A \wedge B) \wedge (A \Rightarrow B) \Rightarrow B.$$

But so does the following:

$$\dfrac{\dfrac{\dfrac{[(A \wedge B) \wedge (A \Rightarrow B)]^1}{A \Rightarrow B} \qquad \dfrac{\dfrac{[(A \wedge B) \wedge (A \Rightarrow B)]^1}{A \wedge B}}{A}}{B}}{(A \wedge B) \wedge (A \Rightarrow B) \Rightarrow B}\,{\scriptstyle (1)}$$

As does:

$$\dfrac{\dfrac{\dfrac{[(A \wedge B) \wedge (A \Rightarrow B)]^1}{A \wedge B}}{B}}{(A \wedge B) \wedge (A \Rightarrow B) \Rightarrow B}\,{\scriptstyle (1)}$$

There is a sense in which the first two proofs are "equivalent", but not the first and the third. The relation (or property) of *provability* in propositional calculus $\vdash A$ discards such differences in the proofs that witness it. According to the "proof-relevant" point of view, sometimes called *propositions as types*, one retains as relevant some information about the way in which a proposition is proved. This can be done by annotating the proofs with *proof-terms* as they are constructed, as follows:

$$\dfrac{\dfrac{\dfrac{[x : (A \wedge B) \wedge (A \Rightarrow B)]^1}{\pi_2(x) : A \Rightarrow B} \qquad \dfrac{\dfrac{[x : (A \wedge B) \wedge (A \Rightarrow B)]^1}{\pi_1(x) : A \wedge B}}{\pi_1(\pi_1(x)) : A}}{\pi_2(x)(\pi_1(\pi_1(x))) : B}}{\lambda x.\pi_2(x)(\pi_1(\pi_1(x))) : (A \wedge B) \wedge (A \Rightarrow B) \Rightarrow B}\;(1)$$

$$\dfrac{\dfrac{\dfrac{\dfrac{[x : (A \wedge B) \wedge (A \Rightarrow B)]^1}{\pi_1(x) : A \wedge B}}{\pi_1(\pi_1(x)) : A} \qquad \dfrac{[x : (A \wedge B) \wedge (A \Rightarrow B)]^1}{\pi_2(x) : A \Rightarrow B}}{\pi_2(x)(\pi_1(\pi_1(x))) : B}}{\lambda x.\pi_2(x)(\pi_1(\pi_1(x))) : (A \wedge B) \wedge (A \Rightarrow B) \Rightarrow B}\;(1)$$

$$\dfrac{\dfrac{\dfrac{[x : (A \wedge B) \wedge (A \Rightarrow B)]^1}{\pi_1(x) : A \wedge B}}{\pi_2(\pi_1(x)) : B}}{\lambda x.\pi_2(\pi_1(x)) : (A \wedge B) \wedge (A \Rightarrow B) \Rightarrow B}\;(1)$$

The proof terms for the first two proofs are the same, namely $\lambda x.\pi_2(x)(\pi_1(\pi_1(x)))$, but the term for the third one is $\lambda x.\pi_2(\pi_1(x))$, reflecting the difference in the proofs. The assignment works by labelling assumptions as variables, and then associating term-constructors to the different rules of inference: pairing and projection to conjunction introduction and elimination, function application and λ-abstraction to implication elimination (*modus ponens*) and introduction. The use of variable binding to represent cancellation of premisses is a particularly effective device.

From the categorical point of view, the relation of deducibility $A \vdash B$ is a mere preorder. The addition of proof terms $x : A \vdash t : B$ results in a *categorification* of this preorder, in the sense that it becomes a "proper" category, the preordered reflection of which is the deducibility preorder. And now a remarkable fact emerges: it is hardly surprising that the deducibility preorder has, say, finite products $A \wedge B$ or even exponentials $A \Rightarrow B$; but it is *amazing* that the category with proof terms $x : A \vdash t : B$ as arrows also turns out to be a cartesian closed category, and indeed a proper one, with distinct parallel arrows, such as

$$\begin{aligned} \pi_2(x)(\pi_1(\pi_1(x))) &: (A \wedge B) \wedge (A \Rightarrow B) \longrightarrow B, \\ \pi_2(\pi_1(x)) &: (A \wedge B) \wedge (A \Rightarrow B) \longrightarrow B. \end{aligned}$$

This *category of proofs* contains information about the "proof theory" of the propositional calculus, as opposed to its mere relation of deducibility.

And now another remarkable fact emerges: when the calculus of proof terms is formulated as a system of *simple type theory*, it admits an alternate interpretation as a formal system of *function abstraction and application.* This dual interpretation of the system of type theory—as the proof theory of propositional logic, and as formal system for manipulating functions—is sometimes called the *Curry-Howard correspondence* [Sco70, ML84, Tai68]. From the categorical point of view, it expresses a structural equivalence between the cartesian closed categories of *proofs* in propositional logic and *terms* in simple type theory, both of which are categorifications of their common preorder reflection, the deducibility preorder of propositional logic (cf. [MH92]).

In the following sections, we shall consider this remarkable correspondence in detail, as well as some extensions of the basic case represented by cartesian closed categories: categories with coproducts, cocomplete categories, and categories equipped with modal operators. In the next chapter, it will be seen that this correspondence even extends to proofs in quantified predicate logic and terms in dependent type theory, and beyond.

4.2 Cartesian closed categories

Exponentials

We begin with the notion of an exponential B^A of two objects A, B in a category, motivated by a couple of important examples. Consider first the category Pos of posets and monotone functions. For posets P and Q the set $\mathsf{Hom}(P, Q)$ of all monotone functions between them is again a poset, with the pointwise order:

$$f \leq g \iff fx \leq gx \quad \text{for all } x \in P\ . \qquad (f, g : P \to Q)$$

Thus, when equipped with a suitable order, the *set* $\mathsf{Hom}(P, Q)$ becomes an object of Pos.

Similarly, given monoids $K, M \in \mathsf{Mon}$, there is a natural monoid structure on the set $\mathsf{Hom}(K, M)$, defined pointwise by

$$(f \cdot g)x = fx \cdot gx\ . \qquad (f, g : K \to M,\ x \in K)$$

Thus the category Mon also admits such "internal Homs". The same thing works in the category Group of groups and group homomophisms, where the set $\mathsf{Hom}(G, H)$ of all homomorphisms between groups G and H can be given a pointwise group structure.

These examples suggest a general notion of an "internal Hom" in a category: an "object of morphisms $A \to B$" which corresponds to the hom-set $\mathsf{Hom}(A, B)$. The other ingredient needed is an "evaluation" operation $\mathsf{eval} : B^A \times A \to B$ which evaluates a morphism $f \in B^A$ at an argument $a \in A$ to give a value $\mathsf{eval} \circ \langle f, a \rangle = f(a) \in B$. This is always going to be present as an operation on underlying sets, if we're starting from a set of functions $\mathsf{Hom}(A, B)$ between structured sets A and B, but even in that case it also needs to be an actual morphism in the category. Finally, we need an operation of "transposition", taking a morphism $f : C \times A \to B$ to one $\widetilde{f} : C \to A^B$. We shall see that this in fact separates the previous two examples.

Definition 4.2.1. In a category $\mathcal{C}$ with binary products, an *exponential* (B^A, ϵ) of objects A and B is an object B^A together with a morphism $\epsilon : B^A \times A \to B$, called the *evaluation* morphism, such that for every $f : C \times A \to B$ there exists a *unique* morphism $\widetilde{f} : C \to B^A$, called the *transpose*[1] of f, for which the following diagram commutes.

$$\begin{tikzcd}
B^A & B^A \times A \arrow[r, "\epsilon"] & B \\
C \arrow[u, dotted, "\widetilde{f}"] & C \times A \arrow[u, "\widetilde{f} \times 1_A"] \arrow[ur, "f"'] &
\end{tikzcd}$$

Commutativity of the diagram of course means that $\epsilon \circ (\widetilde{f} \times 1_A) = f$.

Definition 4.2.1 is called the *universal property of the exponential.* It is just the category-theoretic way of saying that a function $f : C \times A \to B$ of two variables can be viewed as a function $\widetilde{f} : C \to B^A$ of one variable that maps $z \in C$ to a function $\widetilde{f}z = f\langle z, -\rangle : A \to B$ that maps $x \in A$ to $f\langle z, x\rangle$. The relationship between f and $\widetilde{f}$ is then the expected one:

$$(\widetilde{f}z)x = f\langle z, x\rangle \ .$$

That is all there is to it, except that by making the evaluation explicit, variables and elements never need to be mentioned! The benefit of this is that the definition makes sense also in categories whose objects are not *sets*, and whose morphisms are not *functions*—even though some of the basic examples are of that sort.

In Poset the exponential Q^P of posets P and Q is the set of all monotone maps $P \to Q$, ordered pointwise, as above. The evaluation map $\epsilon : Q^P \times P \to Q$ is just the usual evaluation of a function at an argument. The transpose of a monotone map $f : R \times P \to Q$ is the map $\widetilde{f} : R \to Q^P$, defined by, $(\widetilde{f}z)x = f\langle z, x\rangle$, i.e. the transposed *function.* We say that the category Pos *has all exponentials.*

Definition 4.2.2. Suppose $\mathcal{C}$ has all finite products. An object $A \in \mathcal{C}$ is *exponentiable* when the exponential B^A exists for every $B \in \mathcal{C}$ (along with an associated evaluation map $\epsilon : B^A \times A \to B$). We say that $\mathcal{C}$ *has exponentials* if every object is exponentiable. A *cartesian closed category (ccc)* is a category that has all finite products and exponentials.

Example 4.2.3. Consider again the example of the set $\mathsf{Hom}(M, N)$ of homomorphisms between two monoids M, N, equipped with the pointwise monoid structure. To be a monoid homomorphism, the transpose $\widetilde{h} : 1 \to \mathsf{Hom}(M, N)$ of a homomorphism $h : 1 \times M \to N$ would have to take the unit element $u \in 1$ to the unit homomorphism $u : M \to N$, which is the constant function at the unit $u \in N$. Since $1 \times M \cong M$, that would mean that *all* homomorphisms $h : M \to N$ would have the same transpose, namely $\widetilde{h} = u : 1 \to \mathsf{Hom}(M, N)$. So Mon cannot be cartesian closed. The same argument works in the category Group, and in many related ones.

[1]Also, f is called the transpose of $\widetilde{f}$, so that f and $\widetilde{f}$ are each other's transpose.

Exercise 4.2.4. Recall that monoids and groups can be regarded as (1-object) categories, and then their homomorphisms are just functors. So we have full subcategories,

$$\mathsf{Mon} \hookrightarrow \mathsf{Group} \hookrightarrow \mathsf{Cat}\,.$$

Is the category Cat of all (small) categories and functors cartesian closed? What about the subcategory of all *groupoids*,

$$\mathsf{Grpd} \hookrightarrow \mathsf{Cat}\,,$$

defined as those categories in which every arrow is an iso?

Two characterizations of CCCs

Proposition 4.2.5. *In a category $\mathcal{C}$ with binary products an object A is exponentiable if, and only if, the functor*

$$- \times A : \mathcal{C} \to \mathcal{C}$$

has a right adjoint

$$-^A : \mathcal{C} \to \mathcal{C}\ .$$

Proof. If such a right adjoint exists then the exponential of A and B is (B^A, ϵ_B), where $\epsilon_B : B^A \times A \to A$ is the counit of the adjunction at B. Indeed, the universal property of the exponential is just the universal property of the counit $\epsilon : (-)^A \Rightarrow 1_{\mathcal{C}}$.

Conversely, suppose for every B there is an exponential (B^A, ϵ_B). As the object part of the right adjoint we then take B^A. For the morphism part, given $g : B \to C$, we can define $g^A : B^A \to C^A$ to be the transpose of $g \circ \epsilon_B$,

$$g^A = (g \circ \epsilon_B)^\sim$$

as indicated below.

$$\begin{tikzcd}
B^A \times A \arrow[r, "\epsilon_B"] \arrow[d, "g^A \times 1_A"'] & B \arrow[d, "g"] \\
C^A \times A \arrow[r, "\epsilon_C"'] & C
\end{tikzcd} \tag{4.1}$$

The counit $\epsilon : -^A \times A \Longrightarrow 1_{\mathcal{C}}$ at B is then ϵ_B itself, and the naturality square for ϵ is then exactly (4.1), i.e. the defining property of $(f \circ \epsilon_B)^\sim$:

$$\epsilon_C \circ (g^A \times 1_A) = \epsilon_C \circ ((g \circ \epsilon_B)^\sim \times 1_A) = g \circ \epsilon_B\ .$$

The universal property of the counit ϵ is precisely the universal property of the exponential (B^A, ϵ_B) □

Note that because exponentials can be expressed as right adjoints to binary products, they are determined uniquely up to isomorphism. Moreover, the definition of a cartesian closed category can then be phrased entirely in terms of adjoint functors: we just need to require the existence of the terminal object, binary products, and exponentials.

Proposition 4.2.6. *A category $\mathcal{C}$ is cartesian closed if, and only if, the following functors have* right *adjoints:*

$$
\begin{aligned}
!_{\mathcal{C}} &: \mathcal{C} \to 1 \;, \\
\Delta &: \mathcal{C} \to \mathcal{C} \times \mathcal{C} \;, \\
(- \times A) &: \mathcal{C} \to \mathcal{C} \;. && (A \in \mathcal{C})
\end{aligned}
$$

Here $!_{\mathcal{C}}$ is the unique functor from $\mathcal{C}$ to the terminal category 1 and Δ is the diagonal functor $\Delta A = \langle A, A \rangle$, and the right adjoint of $- \times A$ is exponentiation by A.

□

The significance of the adjoint formulation is that it implies the possibility of a purely *equational* specification (adjoint structure on a category is "algebraic", in a sense that can be made precise; see [**?**]). It follows that there is a equational formulation of the definition of a cartesian closed category.

Proposition 4.2.7 (Equational version of CCC)**.** *A category $\mathcal{C}$ is cartesian closed if, and only if, it has the following structure:*

1. *An object $1 \in \mathcal{C}$ and a morphism $!_A : A \to 1$ for every $A \in \mathcal{C}$.*
2. *An object $A \times B$ for all $A, B \in \mathcal{C}$ together with morphisms $\pi_0 : A \times B \to A$ and $\pi_1 : A \times B \to B$, and for every pair of morphisms $f : C \to A$, $g : C \to B$ a morphism $\langle f, g \rangle : C \to A \times B$.*
3. *An object B^A for all $A, B \in \mathcal{C}$ together with a morphism $\epsilon : B^A \times A \to B$, and a morphism $\widetilde{f} : C \to B^A$ for every morphism $f : C \times A \to B$.*

These new objects and morphisms are required to satisfy the following equations:

1. *For every $f : A \to 1$,*
$$f = \,!_A \;.$$
2. *For all $f : C \to A$, $g : C \to B$, $h : C \to A \times B$,*
$$\pi_0 \circ \langle f, g \rangle = f \;, \qquad \pi_1 \circ \langle f, g \rangle = g \;, \qquad \langle \pi_0 \circ h, \pi_1 \circ h \rangle = h \;.$$
3. *For all $f : C \times A \to B$, $g : C \to B^A$,*
$$\epsilon \circ (\widetilde{f} \times 1_A) = f \;, \qquad (\epsilon \circ (g \times 1_A))^{\sim} = g \;.$$
where for $e : E \to E'$ and $f : F \to F'$ we define
$$e \times f := \langle e\pi_0, f\pi_1 \rangle : E \times F \to E' \times F'.$$

These equations ensure that certain diagrams commute and that the morphisms that are required to exist are unique. For example, let us prove that $(A \times B, \pi_0, \pi_1)$ is the product of A and B. For $f : C \to A$ and $g : C \to B$ there exists a morphism $\langle f, g\rangle : C \to A \times B$. Equations

$$\pi_0 \circ \langle f, g\rangle = f \qquad \text{and} \qquad \pi_1 \circ \langle f, g\rangle = g$$

enforce the commutativity of the two triangles in the following diagram:

$$\begin{tikzcd} & C \arrow[dl, "g"'] \arrow[d, "\langle f,g\rangle"] \arrow[dr, "f"] & \\ A & A \times B \arrow[l, "\pi_0"] \arrow[r, "\pi_1"'] & B \end{tikzcd}$$

Suppose $h : C \to A \times B$ is another morphism such that $f = \pi_0 \circ h$ and $g = \pi_1 \circ h$. Then by the third equation for products we get

$$h = \langle \pi_0 \circ h, \pi_1 \circ h\rangle = \langle f, g\rangle \ ,$$

and so $\langle f, g\rangle$ is unique.

Exercise 4.2.8. Use the equational characterization of CCCs, Proposition 4.2.7, to show that the category Pos of posets and monotone functions *is* cartesian closed, as claimed. Also verify that that Mon is not. Which parts of the definition fail in Mon?

Exercise 4.2.9. Use the equational characterization of CCCs, Proposition 4.2.7, to show that the product category $\Pi_{i\in I}\,\mathcal{C}_i$ of any (set-indexed) family $(\mathcal{C}_i)_{i\in I}$ of cartesian closed categories $\mathcal{C}_i$ is cartesian closed. Is the same true for an arbitrary limit in Cat?

Some proper CCCs

We next review some important examples of (non-poset) cartesian closed categories, most of which have already been discussed.

Example 4.2.10. The first example is the category Set. We already know that the terminal object is a singleton set and that binary products are cartesian products. The exponential of X and Y in Set is just the set of all functions from X to Y,

$$Y^X = \big\{ f \subseteq X \times Y \;\big|\; \forall x : X \,.\, \exists!\, y : Y \,.\, \langle x, y\rangle \in f \big\} \ .$$

The evaluation morphism $\mathsf{eval} : Y^X \times X \to Y$ is the usual evaluation of a function at an argument, i.e., $\mathsf{eval}\langle f, x\rangle$ is the unique $y \in Y$ for which $\langle x, y\rangle \in f$.

Example 4.2.11. The category Cat of all small categories is cartesian closed. The exponential of small categories $\mathcal{C}$ and $\mathcal{D}$ is the category $\mathcal{D}^{\mathcal{C}}$ of functors, with natural transformations as arrows (see **??**). Note that if $\mathcal{D}$ is a groupoid (all arrows are isos), then so is $\mathcal{D}^{\mathcal{C}}$. It follows that the category of groupoids is full (even as a 2-category) in Cat. Since limits of groupoids in Cat are also groupoids, the inclusion of the full subcategory $\mathsf{Grpd} \hookrightarrow \mathsf{Cat}$ preserves limits. It also preserves the CCC structure.

Example 4.2.12. The same reasoning as in the previous example shows that the full subcategory $\mathsf{Pos} \hookrightarrow \mathsf{Cat}$ of all small posets and monotone maps is also cartesian closed, and the (limit preserving) inclusion $\mathsf{Pos} \hookrightarrow \mathsf{Cat}$ also preserves exponentials. Note that the (non-full) forgetful functor $U : \mathsf{Pos} \to \mathsf{Set}$ does not, and that $U(Q^P) \subseteq (UQ)^{UP}$ is in general a *proper* subset.

Exercise 4.2.13. There is a full and faithful functor $I : \mathsf{Set} \to \mathsf{Poset}$ that preserves finite limits as well as exponentials. How is this related to the example $\mathsf{Grpd} \hookrightarrow \mathsf{Cat}$?

The foregoing examples are instances of the following general situation.

Proposition 4.2.14. *Let $\mathcal{E}$ be a CCC and $i : \mathcal{S} \hookrightarrow \mathcal{E}$ a full subcategory with finite products and a left adjoint reflection $L : \mathcal{E} \to \mathcal{S}$ preserving finite products. Suppose moreover that for any two objects A, B in $\mathcal{S}$, the exponential iB^{iA} is again in $\mathcal{S}$. Then $\mathcal{S}$ has all exponentials, and these are preserved by i.*

Proof. By assumption, we have $L \dashv i$ with isomorphic counit $LiS \cong S$ for all $S \in \mathcal{S}$. Let us identify $\mathcal{S}$ with the subcategory of $\mathcal{E}$ that is its image under $i : \mathcal{S} \hookrightarrow \mathcal{E}$. The assumption that B^A is again in $\mathcal{S}$ for all $A, B \in \mathcal{S}$, along with the fullness of $\mathcal{S}$ in $\mathcal{E}$, gives the exponentials, and the closure of $\mathcal{S}$ under finite products in $\mathcal{E}$ ensures that the required transposes will also be in $\mathcal{S}$.

Alternately, for any $A, B \in \mathcal{S}$ set $B^A = L(iB^{iA})$. Then for any $C \in \mathcal{S}$, we have natural isos:

$$\begin{aligned}
\mathcal{S}(C \times A, B) &\cong \mathcal{E}(i(C \times A), iB) \\
&\cong \mathcal{E}(iC \times iA, iB) \\
&\cong \mathcal{E}\big(iC, iB^{iA}\big) \\
&\cong \mathcal{E}\big(iC, iL(iB^{iA})\big) \\
&\cong \mathcal{S}\big(C, L(iB^{iA})\big) \\
&\cong \mathcal{S}\big(C, B^A\big)
\end{aligned}$$

where in the fifth line we used the assumption that iB^{iA} is again in $\mathcal{S}$, in the form $iB^{iA} \cong iE$ for some $E \in \mathcal{S}$, which is then necessarily $L(iB^{iA}) = LiE \cong E$. □

A related general situation that covers some (but not all) of the above examples is this:

Proposition 4.2.15. *Let $\mathcal{E}$ be a CCC and $i : \mathcal{S} \hookrightarrow \mathcal{E}$ a full subcategory with finite products and a right adjoint reflection $R : \mathcal{E} \to \mathcal{S}$. If i preserves finite products, then $\mathcal{S}$ also has all exponentials, and these are computed first in $\mathcal{E}$, and then reflected by R into $\mathcal{S}$.*

Proof. For any $A, B \in \mathcal{S}$ set $B^A = R(iB^{iA})$ as described. Now for any $C \in \mathcal{S}$, we have natural isos:

$$\begin{aligned}
\mathcal{S}(C \times A, B) &\cong \mathcal{E}(i(C \times A), iB) \\
&\cong \mathcal{E}(iC \times iA, iB) \\
&\cong \mathcal{E}\big(iC, iB^{iA}\big) \\
&\cong \mathcal{S}\big(C, R(iB^{iA})\big) \\
&\cong \mathcal{S}\big(C, B^A\big) .
\end{aligned}$$

□

An example of the foregoing is the inclusion of the opens into the powerset of points of a space X,

$$\mathcal{O}X \hookrightarrow \mathcal{P}X$$

This frame homomorphism is associated to the map $|X| \to X$ of locales (or in this case, spaces) from the discrete space on the set of points of X.

Exercise 4.2.16. Which of the examples follows from which proposition?

Example 4.2.17. For any set X, the slice category $\mathsf{Set}/_X$ is cartesian closed. The product of $f : A \to X$ and $g : B \to X$ is the pullback $A \times_X B \to X$, which can be constructed as the set of pairs

$$A \times_X B \to X = \{\langle a, b\rangle \mid fa = gb\} .$$

The exponential, however, is *not* simply the set

$$\{h : A \to B \mid f = g \circ h\} ,$$

(what would the projection to X be?), but rather the set of all pairs

$$\{\langle x, h : A_x \to B_x\rangle \mid x \in X,\ f = g \circ h\} ,$$

where $A_x = f^{-1}\{x\}$ and $B_x = g^{-1}\{x\}$, with the evident projection to X.

Exercise 4.2.18. Prove that $\mathsf{Set}/_X$ is always cartesian closed.

Example 4.2.19. A presheaf category $\widehat{\mathbb{C}}$ is cartesian closed, provided the index category $\mathbb{C}$ is small. To see what the exponential of presheaves P and Q ought to be, we use the Yoneda Lemma. If Q^P exists, then by Yoneda Lemma and the adjunction $(- \times P) \dashv (-^P)$, we have for all $A \in \mathbb{C}$,

$$Q^P(A) \cong \mathsf{Nat}(\mathsf{y}A, Q^P) \cong \mathsf{Nat}(\mathsf{y}A \times P, Q) .$$

Because $\mathcal{C}$ is small $\mathsf{Nat}(\mathsf{y}A \times P, Q)$ is a set, so we can *define* Q^P to be the presheaf

$$Q^P = \mathsf{Nat}(\mathsf{y}- \times P, Q) .$$

The evaluation morphism $E : Q^P \times P \Longrightarrow Q$ is the natural transformation whose component at A is

$$E_A : \mathsf{Nat}(\mathsf{y}A \times P, Q) \times PA \to QA\;,$$
$$E_A : \langle \eta, x \rangle \mapsto \eta_A \langle 1_A, x \rangle\;.$$

The transpose of a natural transformation $\phi : R \times P \Longrightarrow Q$ is the natural transformation $\widetilde{\phi} : R \Longrightarrow Q^P$ whose component at A is the function that maps $z \in RA$ to the natural transformation $\widetilde{\phi}_A z : \mathsf{y}A \times P \Longrightarrow Q$, whose component at $B \in \mathcal{C}$ is

$$(\widetilde{\phi}_A z)_B : \mathcal{C}(B, A) \times PB \to QB\;,$$
$$(\widetilde{\phi}_A z)_B : \langle f, y \rangle \mapsto \phi_B \langle (Rf) z, y \rangle\;.$$

Exercise 4.2.20. Verify that the above definition of Q^P really gives an exponential of presheaves P and Q.

It follows immediately that the category of graphs Graph is cartesian closed because it is the presheaf category $\mathsf{Set}^{\cdot \rightrightarrows \cdot}$. The same is of course true for the "category of functions", i.e. the arrow category $\mathsf{Set}^{\to}$, as well as the category of simplicial sets $\mathsf{Set}^{\Delta^{\mathsf{op}}}$ from topology.

Exercise 4.2.21. This exercise is for students with some background in linear algebra. Let Vec be the category of real vector spaces and linear maps between them. Given vector spaces X and Y, the linear maps $\mathcal{L}(X, Y)$ between them form a vector space. So define $\mathcal{L}(X, -) : \mathsf{Vec} \to \mathsf{Vec}$ to be the functor which maps a vector space Y to the vector space $\mathcal{L}(X, Y)$, and it maps a linear map $f : Y \to Z$ to the linear map $\mathcal{L}(X, f) : \mathcal{L}(X, Y) \to \mathcal{L}(X, Z)$ defined by $h \mapsto f \circ h$. Show that $\mathcal{L}(X, -)$ has a left adjoint $- \otimes X$, but also show that this adjoint is *not* the binary product in Vec.

A few other instructive examples that can be explored by the interested reader are the following.

- Etale spaces over a base space X. This category can be described as consisting of *local homeomorphisms* $f : Y \to X$ and commutative triangles over X between such maps. It is equivalent to the category $\mathsf{Sh}(X)$ of *sheaves* on X. See [**?**, ch.n].

- Various subcategories of topological spaces (sequential spaces, compactly-generated spaces). Cf. [**?**].

- Dana Scott's category Equ of equilogical spaces [**?**].

4.3 Simple type theory

The λ-calculus is an abstract theory of functions, much like group theory is an abstract theory of symmetries. There are two basic operations that can be performed with functions. The first one is the *application* of a function to an argument: if f is a function and a is an argument, then fa is the application of f to a, also called the *value* of f at a. The second operation is *abstraction*: if x is a variable and t is an expression in which x may appear, then there is a function f defined by the equation

$$fx = t \; .$$

Here we gave the name f to the newly formed function. But we could have expressed the same function without giving it a name; this is usually written as

$$x \mapsto t \; ,$$

and it means "x is mapped to t". In λ-calculus we use a different notation, which is more convenient when such abstractions are nested, namely

$$\lambda x.\, t \; .$$

This operation is called λ-*abstraction*. For example, $\lambda x.\, \lambda y.\, (x + y)$ is the function that maps an argument a to the function $\lambda y.\, (a + y)$, which maps an argument b to the value $a + b$. The variable x is said to be *bound in* t in the expression $\lambda x.\, t$.

It may seem strange that in specifying the abstraction of a function, we switched from talking about objects (functions, arguments, values) to talking about *expressions*: variables, names, equations. This "syntactic" point of view seems to have been part of the notion of a function from the start, in the theory of algebraic equations. It is the reason that the λ-calculus is part of *logic*, unlike the theory of cartesian closed categories, which remains thoroughly semantical (and "variable-free"). The relation between the two different points of view occupies the rest of this chapter—and, indeed, the entire subject of logic!

There are two kinds of λ-calculus: the *typed* and the *untyped*. In the untyped version there are no restrictions on how application is formed, so that an expression such as

$$\lambda x.\, (xx)$$

is valid, whatever it may mean. We will concentrate here on the typed λ-calculus. In typed λ-calculus every expression has a *type*, and there are rules for forming valid expressions and types. For example, we can only form an application fa when a has a type A and f has a type $A \to B$, which indicates a function taking arguments of type A and giving results of type B. The judgment that expression t has a type A is written as

$$t : A \; .$$

To computer scientists the idea of expressions having types is familiar from programming languages, whereas mathematicians can think of types as sets and read $t : A$ as $t \in A$.

Simply-typed λ-calculus. We now give a more formal definition of what constitutes a *simply-typed λ-calculus.* First, we are given a set of *simple types*, which are generated from *basic types* by formation of products and function types:

$$\begin{array}{rl} \text{Basic types} & \mathtt{B} ::= \mathtt{B}_0 \mid \mathtt{B}_1 \mid \mathtt{B}_2 \cdots \\ \text{Simple types} & A ::= \mathtt{B} \mid A_1 \times A_2 \mid A_1 \to A_2. \end{array}$$

We adopt the convention that function types associate to the right:

$$A \to B \to C \;=\; A \to (B \to C)\;.$$

We assume there is a countable set of *variables* x, y, u, ... We are also given a set of *basic constants.* The set of *terms* is generated from variables and basic constants by the following grammar:

$$\begin{array}{rl} \text{Variables} & v ::= x \mid y \mid z \mid \cdots \\ \text{Constants} & c ::= \mathtt{c}_1 \mid \mathtt{c}_2 \mid \cdots \\ \text{Terms} & t ::= v \mid c \mid * \mid \langle t_1, t_2 \rangle \mid \mathtt{fst}\, t \mid \mathtt{snd}\, t \mid t_1\, t_2 \mid \lambda x : A\,.\,t \end{array}$$

In words, this means:

1. a variable is a term,
2. each basic constant is a term,
3. the constant $*$ is a term, called the *unit*,
4. if u and t are terms then $\langle u, t\rangle$ is a term, called a *pair*,
5. if t is a term then $\mathtt{fst}\, t$ and $\mathtt{snd}\, t$ are terms,
6. if u and t are terms then $u\,t$ is a term, called an *application*
7. if x is a variable, A is a type, and t is a term, then $\lambda x : A\,.\,t$ is a term, called a λ-*abstraction.*

The variable x is *bound* in $\lambda x : A\,.\,t$. Application associates to the left, thus $s\,t\,u = (s\,t)\,u$. The set of *free variables* $\mathsf{FV}(t)$ of a term t is determined as follows:

$$\begin{aligned} \mathsf{FV}(x) &= \{x\} \quad \text{if } x \text{ is a variable} \\ \mathsf{FV}(a) &= \emptyset \qquad \text{if } a \text{ is a basic constant} \\ \mathsf{FV}(\langle u, t\rangle) &= \mathsf{FV}(u) \cup \mathsf{FV}(t) \\ \mathsf{FV}(\mathtt{fst}\, t) &= \mathsf{FV}(t) \\ \mathsf{FV}(\mathtt{snd}\, t) &= \mathsf{FV}(t) \\ \mathsf{FV}(u\, t) &= \mathsf{FV}(u) \cup \mathsf{FV}(t) \\ \mathsf{FV}(\lambda x.\, t) &= \mathsf{FV}(t) \setminus \{x\}\;. \end{aligned}$$

If x_1, …, x_n are *distinct* variables and A_1, …, A_n are types then the sequence

$$x_1 : A_1, \ldots, x_n : A_n$$

is a *typing context*, or just *context*. The empty sequence is sometimes denoted by a dot $\cdot$, and it is a valid context. Contexts are denoted by capital Greek letters Γ, Δ, …

A *typing judgment* is a judgment of the form

$$\Gamma \mid t : A$$

where Γ is a context, t is a term, and A is a type. In addition the free variables of t must occur in Γ, but Γ may contain other variables as well. We read the above judgment as "in context Γ the term t has type A". Next we describe the rules for deriving typing judgments.

- Each basic constant $\mathtt{c}_i$ has a uniquely determined type C_i (not necessarily basic):

$$\frac{}{\Gamma \mid \mathtt{c}_i : C_i}$$

- The type of a variable is determined by the context:

$$\frac{}{x_1 : A_1, \ldots, x_i : A_i, \ldots, x_n : A_n \mid x_i : A_i} \ (1 \leq i \leq n)$$

- The constant $*$ has type $\mathtt{1}$:

$$\frac{}{\Gamma \mid * : \mathtt{1}}$$

- The typing rules for pairs and projections are:

$$\frac{\Gamma \mid a : A \qquad \Gamma \mid b : B}{\Gamma \mid \langle a, b\rangle : A \times B} \qquad \frac{\Gamma \mid t : A \times B}{\Gamma \mid \mathtt{fst}\, t : A} \qquad \frac{\Gamma \mid c : A \times B}{\Gamma \mid \mathtt{snd}\, t : B}$$

- The typing rules for application and λ-abstraction are:

$$\frac{\Gamma \mid t : A \to B \qquad \Gamma \mid a : A}{\Gamma \mid t\, a : B} \qquad \frac{\Gamma, x : A \mid t : B}{\Gamma \mid (\lambda x : A \,.\, t) : A \to B}$$

Lastly, we have *equations* between terms: for terms of type A in context Γ,

$$\Gamma \mid s : A\,, \qquad\qquad \Gamma \mid t : A\,,$$

the judgment that they are equal is written as

$$\Gamma \mid s = t : A\,.$$

Note that s and t necessarily have the same type; it does *not* make sense to compare terms of different types. We have the following rules for equations, the effect of which is to make equality between terms into an equivalence relation at each type, and a congruence with respect to all of the operations, just as for algebraic theories:

- Equality is an equivalence relation:

$$\frac{}{\Gamma \mid t = t : A} \qquad \frac{\Gamma \mid s = t : A}{\Gamma \mid t = s : A} \qquad \frac{\Gamma \mid s = t : A \qquad \Gamma \mid t = u : A}{\Gamma \mid s = u : A}$$

- The substitution rule:

$$\frac{\Gamma \mid s = t : A \qquad \Gamma, x : A \mid u = v : B}{\Gamma \mid u[s/x] = v[t/x] : B}$$

- The weakening rule:

$$\frac{\Gamma \mid s = t : A}{\Gamma, x : B \mid s = t : A}$$

- Unit type:

$$\frac{}{\Gamma \mid t = * : \mathtt{1}}$$

- Equations for product types:

$$\frac{\Gamma \mid u = v : A \qquad \Gamma \mid s = t : B}{\Gamma \mid \langle u, s\rangle = \langle v, t\rangle : A \times B}$$

$$\frac{\Gamma \mid s = t : A \times B}{\Gamma \mid \mathtt{fst}\, s = \mathtt{fst}\, t : A} \qquad \frac{\Gamma \mid s = t : A \times B}{\Gamma \mid \mathtt{snd}\, s = \mathtt{snd}\, t : A}$$

$$\frac{}{\Gamma \mid t = \langle \mathtt{fst}\, t, \mathtt{snd}\, t\rangle : A \times B}$$

$$\frac{}{\Gamma \mid \mathtt{fst}\, \langle s, t\rangle = s : A} \qquad \frac{}{\Gamma \mid \mathtt{snd}\, \langle s, t\rangle = t : A}$$

- Equations for function types:

$$\frac{\Gamma \mid s = t : A \to B \qquad \Gamma \mid u = v : A}{\Gamma \mid s\, u = t\, v : B}$$

$$\frac{\Gamma, x : A \mid t = u : B}{\Gamma \mid (\lambda x : A \,.\, t) = (\lambda x : A \,.\, u) : A \to B}$$

$$\frac{}{\Gamma \mid (\lambda x : A \,.\, t) u = t[u/x] : A} \qquad (\beta\text{-rule})$$

$$\frac{}{\Gamma \mid \lambda x : A \,.\, (t\, x) = t : A \to B} \text{ if } x \notin \mathsf{FV}(t) \qquad (\eta\text{-rule})$$

This completes the description of a simply-typed λ-calculus.

Simply-typed λ-theories. Apart from the above rules for equality, which are part of the λ-calculus, we might want to impose additional equations between terms. In this case we speak of a *λ-theory.* Thus, a λ-theory $\mathbb{T}$ is given by a set of basic types and a set of basic constants, called the *signature*, and a set of *equations* of the form

$$\Gamma \mid s = t : A \,.$$

Note that we can always state the equations equivalently in *closed form* simply by λ-abstracting all the variables in the context Γ.

We summarize the preceding definitions.

Definition 4.3.1. A *(simply-typed) signature* S is given by a set of *basic types* $(\mathtt{B}_i)_{i\in I}$ together with a set of *basic (typed) constants* $(\mathtt{c}_j : C_j)_{j\in J}$,

$$S \;=\; \big((\mathtt{B}_i)_{i\in I},\, (\mathtt{c}_j : C_j)_{j\in J}\big)\,.$$

A simply-typed λ-theory $\mathbb{T} = (S, E)$ is a simply-typed signature S together with a set of equations between terms,

$$E = \big(u_k = v_k : A_k\big)_{k\in K}\,.$$

Example 4.3.2. The theory of a group is a simply-typed λ-theory. It has one basic type $\mathtt{G}$ and three basic constants, the unit $\mathtt{e}$, the inverse $\mathtt{i}$, and the group operation $\mathtt{m}$,

$$\mathtt{e} : \mathtt{G}\,, \qquad\qquad \mathtt{i} : \mathtt{G} \to \mathtt{G}\,, \qquad\qquad \mathtt{m} : \mathtt{G}\times\mathtt{G} \to \mathtt{G}\,,$$

with the following familiar equations:

$$\begin{gathered}
x : \mathtt{G} \mid \mathtt{m}\langle x, \mathtt{e}\rangle = x : \mathtt{G}\\
x : \mathtt{G} \mid \mathtt{m}\langle \mathtt{e}, x\rangle = x : \mathtt{G}\\
x : \mathtt{G} \mid \mathtt{m}\langle x, \mathtt{i}\,x\rangle = \mathtt{e} : \mathtt{G}\\
x : \mathtt{G} \mid \mathtt{m}\langle \mathtt{i}\,x, x\rangle = \mathtt{e} : \mathtt{G}\\
x : \mathtt{G}, y : \mathtt{G}, z : \mathtt{G} \mid \mathtt{m}\langle x, \mathtt{m}\langle y, z\rangle\rangle = \mathtt{m}\langle \mathtt{m}\langle x, y\rangle, z\rangle : \mathtt{G}
\end{gathered}$$

Example 4.3.3. More generally, any (Lawvere) algebraic theory $\mathbb{A}$ (as in Chapter **??**) determines a λ-theory $\mathbb{A}^\lambda$. There is one basic type $\mathtt{A}$ and for each operation f of arity k there is a basic constant $\mathtt{f} : \mathtt{A}^k \to \mathtt{A}$, where $\mathtt{A}^k$ is the k-fold product $\mathtt{A}\times\cdots\times\mathtt{A}$. It is understood that $\mathtt{A}^0 = \mathtt{1}$. The terms of $\mathbb{A}$ are translated to corresponding terms of $\mathbb{A}^\lambda$ in a straightforward manner. For every axiom $u = v$ of $\mathbb{A}$ there is a corresponding one in $\mathbb{A}^\lambda$,

$$x_1 : \mathtt{A}, \ldots, x_n : \mathtt{A} \mid u = v : \mathtt{A}$$

where $x_1, \ldots, x_n$ are the variables occurring in u and v.

Example 4.3.4. The theory of a directed graph is a simply-typed theory with two basic types, $\mathtt{V}$ for vertices and $\mathtt{E}$ for edges, and two basic constants, source $\mathtt{src}$ and target $\mathtt{trg}$,

$$\mathtt{src} : \mathtt{E} \to \mathtt{V}\,, \qquad\qquad \mathtt{trg} : \mathtt{E} \to \mathtt{V}\,.$$

There are no equations.

Example 4.3.5. The theory of a simplicial set is a simply-typed theory with one basic type $\mathtt{X}_n$ for each natural number n, and the following basic constants, also for each n, and each $0 \leq i \leq n$:

$$\mathtt{d}_i : \mathtt{X}_{n+1} \to \mathtt{X}_n \;, \qquad\qquad \mathtt{s}_i : \mathtt{X}_n \to \mathtt{X}_{n+1} \;.$$

The equations are as follows, for all natural numbers i, j:

$$\begin{aligned}
\mathtt{d}_i \mathtt{d}_j &= \mathtt{d}_{j-1} \mathtt{d}_i, \quad \text{if } i < j, \\
\mathtt{s}_i \mathtt{s}_j &= \mathtt{s}_{j+1} \mathtt{s}_i, \quad \text{if } i \leq j, \\
\mathtt{d}_i \mathtt{s}_j &= \begin{cases} \mathtt{s}_{j-1} \mathtt{d}_i, & \text{if } i < j, \\ \mathsf{id}, & \text{if } i = j \text{ or } i = j+1, \\ \mathtt{s}_j \mathtt{d}_{i-1}, & \text{if } i > j+1. \end{cases}
\end{aligned}$$

Example 4.3.6. An example of a λ-theory found in the theory of programming languages is the mini-programming language *PCF*. It is a theory in simply-typed λ-calculus with a basic type `nat` for natural numbers, and a basic type `bool` of Boolean values,

$$\text{Basic types} \quad \mathtt{B} ::= \mathtt{nat\ type} \mid \mathtt{bool\ type}.$$

There are basic constants zero `0`, successor `succ`, the Boolean constants `true` and `false`, comparison with zero `iszero`, and for each type A the *conditional* $\mathtt{cond}_A$ and the *fixpoint* operator $\mathtt{fix}_A$. They have the following types:

$$\begin{aligned}
\mathtt{0} &: \mathtt{nat} \\
\mathtt{succ} &: \mathtt{nat} \to \mathtt{nat} \\
\mathtt{true} &: \mathtt{bool} \\
\mathtt{false} &: \mathtt{bool} \\
\mathtt{iszero} &: \mathtt{nat} \to \mathtt{bool} \\
\mathtt{cond}_A &: \mathtt{bool} \to A \to A \\
\mathtt{fix}_A &: (A \to A) \to A
\end{aligned}$$

The equational axioms of PCF are:

$$\begin{gathered}
\cdot \mid \mathtt{iszero}\, \mathtt{0} = \mathtt{true} : \mathtt{bool} \\
x : \mathtt{nat} \mid \mathtt{iszero}\, (\mathtt{succ}\, x) = \mathtt{false} : \mathtt{bool} \\
u : A, t : A \mid \mathtt{cond}_A\, \mathtt{true}\; u\; t = u : A \\
u : A, t : A \mid \mathtt{cond}_A\, \mathtt{false}\; u\; t = t : A \\
t : A \to A \mid \mathtt{fix}_A\; t = t\, (\mathtt{fix}_A\; t) : A
\end{gathered}$$

Example 4.3.7 (D.S. Scott)**.** Another example of a λ-theory is the *theory of a reflexive type*. This theory has one basic type `D` and two constants

$$\mathtt{r} : \mathtt{D} \to \mathtt{D} \to \mathtt{D} \qquad\qquad \mathtt{s} : (\mathtt{D} \to \mathtt{D}) \to \mathtt{D}$$

satisfying the equation

$$f : \mathtt{D} \to \mathtt{D} \mid \mathtt{r}\,(\mathtt{s}\,f) = f : \mathtt{D} \to \mathtt{D} \tag{4.2}$$

which says that $\mathtt{s}$ is a section and $\mathtt{r}$ is a retraction, so that the function type $\mathtt{D} \to \mathtt{D}$ is a subspace (even a retract) of $\mathtt{D}$. A type with this property is said to be *reflexive*. We may additionally stipulate the axiom

$$x : \mathtt{D} \mid \mathtt{s}\,(\mathtt{r}\,x) = x : \mathtt{D} \tag{4.3}$$

which implies that $\mathtt{D}$ is isomorphic to $\mathtt{D} \to \mathtt{D}$.

A reflexive type can be used to interpret the untyped λ-calculus into the typed λ-calculus.

Untyped λ-calculus

We briefly describe the *untyped λ-calculus*. It is a theory whose terms are generated by the following grammar:

$$t ::= v \mid t_! \, t_2 \mid \lambda x.\, t\ .$$

In words, a variable is a term, an application $t\,t'$ is a term, for any terms t and t', and a λ-abstraction $\lambda x.\, t$ is a term, for any term t. Variable x is bound in $\lambda x.\, t$. A *context* is a list of distinct variables,

$$x_1, \ldots, x_n\ .$$

We say that a term t is valid in context Γ if the free variables of t are listed in Γ. The judgment that two terms u and t are equal is written as

$$\Gamma \mid u = t\ ,$$

where it is assumed that u and t are both valid in Γ. The context Γ is not really necessary but we include it because it is always good practice to list the free variables.

The rules of equality are as follows:

1. Equality is an equivalence relation:

$$\frac{}{\Gamma \mid t = t} \qquad \frac{\Gamma \mid t = u}{\Gamma \mid u = t} \qquad \frac{\Gamma \mid t = u \qquad \Gamma \mid u = v}{\Gamma \mid t = v}$$

2. The weakening rule:

$$\frac{\Gamma \mid u = t}{\Gamma, x \mid u = t}$$

3. Equations for application and λ-abstraction:

$$\frac{\Gamma \mid s = t \qquad \Gamma \mid u = v}{\Gamma \mid s\,u = t\,v} \qquad \frac{\Gamma, x \mid t = u}{\Gamma \mid \lambda x.\, t = \lambda x.\, u}$$

$$\frac{}{\Gamma \mid (\lambda x.\, t)u = t[u/x]} \tag{β-rule}$$

$$\frac{}{\Gamma \mid \lambda x.\, (t\,x) = t} \quad \text{if } x \notin \mathsf{FV}(t) \tag{η-rule}$$

The untyped λ-calculus can be translated into the theory of a reflexive type from Example 4.3.7. An untyped context Γ is translated to a typed context Γ^* by typing each variable in Γ with the reflexive type $\mathtt{D}$, i.e., a context $x_1, \ldots, x_k$ is translated to $x_1 : \mathtt{D}, \ldots, x_k : \mathtt{D}$. An untyped term t is translated to a typed term t^* as follows:

$$\begin{aligned} x^* &= x \qquad \text{if } x \text{ is a variable} , \\ (u\, t)^* &= (\mathtt{r}\, u^*) t^* , \\ (\lambda x.\, t)^* &= \mathtt{s}\, (\lambda x : \mathtt{D} .\, t^*) . \end{aligned}$$

For example, the term $\lambda x.\, (x\, x)$ translates to $\mathtt{s}\, (\lambda x : \mathtt{D} .\, ((\mathtt{r}\, x)\, x))$. A judgment

$$\Gamma \mid u = t \tag{4.4}$$

is translated to the judgment

$$\Gamma^* \mid u^* = t^* : \mathtt{D} . \tag{4.5}$$

Exercise* 4.3.8. Prove that if equation (4.4) is provable then equation (4.5) is provable as well. Identify precisely at which point in your proof you need to use equations (4.2) and (4.3). Does provability of (4.5) imply provability of (4.4)?

Higher-order logic

This example presumes familiarity with the results of Chapter **??**, or at least with the basic categorical approach to first-order logic as presented in [**?**, **?**]. The approach to IHOL presented here is closely tied to *topos theory*, which is to be treated in greater depth in Chapter **??**.

To be added ...

4.4 Interpretation of λ-calculus in a CCC

We now consider semantic aspects of the λ-calculus and λ-theories. Suppose $\mathbb{T}$ is a λ-theory and $\mathcal{C}$ is a cartesian closed category. An *interpretation* $[\![-]\!]$ of $\mathbb{T}$ in $\mathcal{C}$ is given by the following data:

- For every basic type $\mathtt{B}$ in $\mathbb{T}$ an object $[\![\mathtt{B}]\!] \in \mathcal{C}$. The interpretation is extended to all types by

$$[\![\mathtt{1}]\!] = 1 , \qquad [\![A \times B]\!] = [\![A]\!] \times [\![B]\!] , \qquad [\![A \to B]\!] = [\![B]\!]^{[\![A]\!]} .$$

- For every basic constant $\mathtt{c}$ of type C, a morphism $[\![\mathtt{c}]\!] : 1 \to [\![\mathtt{C}]\!]$.

The interpretation is extended to all terms in context as follows.

- A context $\Gamma = x_1 : A_1, \cdots, x_n : A_n$ is interpreted as the object
$$[\![A_1]\!] \times \cdots \times [\![A_n]\!] \,,$$
and the empty context is interpreted as the terminal object,
$$[\![\cdot]\!] = \mathsf{1}\,.$$
- A typing judgment
$$\Gamma \mid t : A$$
will be interpreted as a morphism
$$[\![\Gamma \mid t : A]\!] : [\![\Gamma]\!] \to [\![A]\!]\,.$$

The interpretation is defined inductively by the following rules:

- The i-th variable is interpreted as the i-th projection,
$$[\![x_0 : A_0, \ldots, x_n : A_n \mid x_i : A_i]\!] = \pi_i : [\![\Gamma]\!] \to [\![A_i]\!]\,.$$
- A basic constant $\mathtt{c} : \mathtt{C}$ in context Γ is interpreted as the composition
$$\begin{tikzcd}[\![\Gamma]\!] \arrow[r, "!_{[\![\Gamma]\!]}"] & \mathsf{1} \arrow[r, "[\![\mathtt{c}]\!]"] & [\![A]\!]\end{tikzcd}$$
- The interpretation of projections and pairs is
$$\begin{aligned}[\![\Gamma \mid \langle t, u\rangle : A \times B]\!] &= \langle [\![\Gamma \mid t : A]\!], [\![\Gamma \mid u : B]\!]\rangle : [\![\Gamma]\!] \to [\![A]\!] \times [\![B]\!]\\ [\![\Gamma \mid \mathtt{fst}\, t : A]\!] &= \pi_0 \circ [\![\Gamma \mid t : A \times B]\!] : [\![\Gamma]\!] \to [\![A]\!]\\ [\![\Gamma \mid \mathtt{snd}\, t : A]\!] &= \pi_1 \circ [\![\Gamma \mid t : A \times B]\!] : [\![\Gamma]\!] \to [\![B]\!]\,.\end{aligned}$$
- The interpretation of application and λ-abstraction is
$$\begin{aligned}[\![\Gamma \mid t\,u : B]\!] &= \epsilon \circ \langle [\![\Gamma \mid t : A \to B]\!], [\![\Gamma \mid u : A]\!]\rangle : [\![\Gamma]\!] \to [\![B]\!]\\ [\![\Gamma \mid \lambda x : A\,.\,t : A \to B]\!] &= ([\![\Gamma, x : A \mid t : B]\!])^{\sim} : [\![\Gamma]\!] \to [\![B]\!]^{[\![A]\!]}\end{aligned}$$
where $\epsilon : [\![A \to B]\!] \times [\![A]\!] \to [\![B]\!]$ is the evaluation morphism for $[\![B]\!]^{[\![A]\!]}$ and $([\![\Gamma, x : A \mid t : B]\!])^{\sim}$ is the transpose of the morphism
$$[\![\Gamma, x : A \mid t : B]\!] : [\![\Gamma]\!] \times [\![A]\!] \to [\![B]\!]\,.$$

Definition 4.4.1. An interpretation of a λ-theory $\mathbb{T}$ is a *model* of $\mathbb{T}$ if it *satisfies* all the axioms of $\mathbb{T}$, in the sense that for every axiom $\Gamma \mid u = v : A$ of $\mathbb{T}$, the interpretations of u and v coincide as arrows in $\mathcal{C}$,
$$[\![\Gamma \mid u : A]\!] = [\![\Gamma \mid v : A]\!] : [\![\Gamma]\!] \longrightarrow [\![A]\!].$$

It follows that all equations that are provable in $\mathbb{T}$ are also satisfied in any model, by the following basic fact.

Proposition 4.4.2 (Soundness). *If $\mathbb{T}$ is a λ-theory and $[\![-]\!]$ a model of $\mathbb{T}$ in a cartesian closed category $\mathcal{C}$, then for every equation in context $\Gamma \mid s = t : C$ that is provable from the axioms of $\mathbb{T}$, we have*

$$[\![\Gamma \mid s : C]\!] = [\![\Gamma \mid t : C]\!] : [\![\Gamma]\!] \longrightarrow [\![C]\!] \,.$$

Briefly, for all $\mathbb{T}$-models $[\![-]\!]$,

$$\mathbb{T} \vdash (\Gamma \mid s = t : C) \quad \textit{implies} \quad [\![-]\!] \models (\Gamma \mid s = t : C) \,.$$

The proof is a straightforward induction, first on the typing judgements for the interpretation, and then on the equational rules for the equations. If we stop after the first step, we can consider just the following notion of *inhabitation*:

Remark 4.4.3 (Inhabitation). There is another notion of provability for the λ-calculus, related to the Curry-Howard correspondence of section 4.1, relating it to propositional logic. If we regard types as "propositions" rather than generalized algebraic structures, and terms as "proofs" rather than operations in such structures, then it is more natural to ask whether there even *is* a term $a : A$ of some type, than whether two terms of the same type are equal $s = t : A$. Of course, this only makes sense when A is considered in the empty context $\cdot \vdash A$, rather than $\Gamma \vdash A$ for non-empty Γ (consider the case where $\Gamma = x : A, \dots$). We say that a type A is *inhabited* (by a closed term) when there is some $\vdash a : A$, and regard an inhabited type A as one that is *provable*. There is then a different notion of soundness related to this notion of provability.

Proposition 4.4.4 (Inhabitation soundness). *If $\mathbb{T}$ is a λ-theory and $[\![-]\!]$ a model of $\mathbb{T}$ in a cartesian closed category $\mathcal{C}$, then for every type A that is inhabited in $\mathbb{T}$, there is a point $1 \to [\![A]\!]$ in $\mathcal{C}$. Thus for all $\mathbb{T}$-models $[\![-]\!]$,*

$$\vdash a : A \quad \textit{implies there is a point} \;\; 1 \to [\![A]\!] \,.$$

This follows immediately from the fact that $[\![\cdot]\!] = 1$ for the empty context; for then the interpretation of any $\vdash a : A$ is a point

$$[\![a]\!] : 1 \to [\![A]\!] \,.$$

Example 4.4.5. 1. A model of an algebraic theory $\mathbb{A}$, extended to a λ-theory $\mathbb{A}^\lambda$ as in Example 4.3.3, taken in a CCC $\mathcal{C}$, is just a model of the algebraic theory $\mathbb{A}$ in the underlying finite product category $|\mathcal{C}|_\times$ of $\mathcal{C}$. An important difference, however, is that in defining the *category of models*

$$\mathsf{Mod}_{\mathsf{FP}}(\mathbb{A}, |\mathcal{C}|_\times)$$

we can take all homomorphisms of models of $\mathbb{A}$ as arrows, while the arrows in the category

$$\mathsf{Mod}_\lambda(\mathbb{A}^\lambda, \mathcal{C})$$

of λ-models are best taken to be isomorphisms, for which one has an obvious way to deal with the contravariance of the function type $[\![A \to B]\!] = [\![B]\!]^{[\![A]\!]}$ (this is discussed in more detail in the next section).

2. A model of the theory of a reflexive type, Example 4.3.7, in Set must be the one-element set $1 = \{\star\}$ (prove this!). Fortunately, the exponentials in categories of presheaves are *not* computed pointwise; otherwise it would follow that this theory has no non-trivial models at all! (And then, by Theorem 4.7.6, that the theory itself is degenerate, in the sense that all equations are provable.) That there are non-trivial models is an important fact in the semantics of programming languages and the subject called *domain theory*. A fundamental paper in which this is shown is [**?**].

3. A (positive) propositional theory $\mathbb{T}$ may be regarded as a λ-theory, and a model in a cartesian closed poset P is then the same thing as before: an interpretation of the atomic propositions $p_1, p_2, \ldots$ of $\mathbb{T}$ as elements $[\![p_1]\!], [\![p_2]\!], \ldots \in P$, such that the axioms $\phi_1, \phi_2, \ldots$ of $\mathbb{T}$ are all sent to $1 \in P$ by the extension of $[\![-]\!]$ to all formulas,

$$1 = [\![\phi_1]\!] = [\![\phi_2]\!] = \cdots \in P\,.$$

Exercise 4.4.6. How are models of a (not necessarily propositional) λ-theory $\mathbb{T}$ in Cartesian closed *posets* related to models in arbitrary Cartesian closed categories? (*Hint:* Consider the inclusion $\mathsf{CCPos} \hookrightarrow \mathsf{CCC}$. Does it have any adjoints?)

4.5 Functorial semantics of STT in CCCs

In Chapter **??** we saw how algebraic theories can be viewed as categories (with finite products), and their algebras, or models, as functors (preserving finite products), and we arranged this analysis of the traditional relationship between syntax and sematics into a framework that we called *functorial semantics*. In Chapter **??**, we did the same for propositional logic. As a common generalization of both, the same framework of functorial semantics can be applied to λ-theories and their models in CCCs. The first step is to build a *classifying category* $\mathcal{C}_{\mathbb{T}}$ from a λ-theory $\mathbb{T}$, which again is constructed from the theory itself as a syntactic category. This is done as follows:

Definition 4.5.1. For any λ-theory $\mathbb{T}$, the *syntactic category* $\mathcal{C}_{\mathbb{T}}$ is determined as follows.

- The objects of $\mathcal{C}_{\mathbb{T}}$ are the types of $\mathbb{T}$.
- Arrows $A \to B$ are terms in context

$$[x : A \mid t : B]\,,$$

  where two such terms $x : A \mid s : B$ and $x : A \mid t : B$ represent the same morphism when $\mathbb{T}$ proves $x : A \mid s = t : B$.

- Composition of the terms

$$[x : A \mid t : B] : A \longrightarrow B \qquad \text{and} \qquad [y : B \mid u : C] : B \longrightarrow C$$

is the term obtained by substituting t for y in u:

$$[x : A \mid u[t/y] : C] : A \longrightarrow C\ .$$

- The identity morphism on A is the term $[x : A \mid x : A]$.

Proposition 4.5.2. *The syntactic category $\mathcal{C}_{\mathbb{T}}$ built from a λ-theory is cartesian closed.*

Proof. We omit the equivalence classes brackets $[x : A \mid t : B]$ and simply treat equivalent terms as equal.

- The terminal object is the unit type $\mathtt{1}$. For any type A the unique morphism $!_A : A \to \mathtt{1}$ is the term

$$x : A \mid * : \mathtt{1}\ .$$

This morphism is indeed unique, because we have the equation

$$\Gamma \mid t = * : \mathtt{1}$$

is an axiom for the terms of unit type $\mathtt{1}$.

- The product of objects A and B is the type $A \times B$. The first and the second projections are the terms

$$c : A \times B \mid \mathtt{fst}\, c : A\ , \qquad\qquad c : A \times B \mid \mathtt{snd}\, c : B\ .$$

Given morphisms

$$z : C \mid a : A\ , \qquad\qquad z : C \mid b : B\ ,$$

the term

$$z : C \mid \langle a, b\rangle : A \times B$$

represents the unique morphism satisfying

$$z : C \mid \mathtt{fst}\, \langle a, b\rangle = a : A\ , \qquad\qquad z : C \mid \mathtt{snd}\, \langle a, b\rangle = b : B\ .$$

Indeed, if $\mathtt{fst}\, t = a$ and $\mathtt{snd}\, t = b$ for some t, then

$$t = \langle \mathtt{fst}\, t, \mathtt{snd}\, t\rangle = \langle a, b\rangle\ .$$

- The exponential of objects A and B is the type $A \to B$ with the evaluation morphism

  $$e : (A \to B) \times A \mid (\mathtt{fst}\, e)(\mathtt{snd}\, e) : B .$$

  The transpose of a morphism $w : C \times A \mid t : B$ is the term

  $$z : C \mid \lambda x : A \,.\, (t[\langle z, x\rangle/w]) : A \to B .$$

  Showing that this is the transpose of t amounts to showing, in context $w : C \times A$,

  $$(\lambda x : A \,.\, (t[\langle \mathtt{fst}\, w, x\rangle/w]))(\mathtt{snd}\, w) = t : B$$

  Indeed, we have:

  $$(\lambda x : A \,.\, (t[\langle \mathtt{fst}\, w, x\rangle/w]))(\mathtt{snd}\, w) = t[\langle \mathtt{fst}\, w, \mathtt{snd}\, w\rangle/w] = t[w/w] = t ,$$

  which is a valid chain of equations in λ-calculus. The transpose is unique, because any morphism $z : C \mid s : A \to B$ that satisfies

  $$(s[\mathtt{fst}\, w/z])(\mathtt{snd}\, w) = t$$

  is equal to $\lambda x : A \,.\, (t[\langle z, x\rangle/w])$, because then

  $$t[\langle z, x\rangle/w] = (s[\mathtt{fst}\, w/z])(\mathtt{snd}\, w)[\langle z, x\rangle/w] = \\ (s[\mathtt{fst}\, \langle z, x\rangle/z])(\mathtt{snd}\, \langle z, x\rangle) = (s[z/z])\, x = s\, x .$$

  Therefore,

  $$\lambda x : A \,.\, (t[\langle z, x\rangle/w]) = \lambda x : A \,.\, (s\, x) = s ,$$

  as claimed.

□

Now as before, the syntactic category allows us to replace a model $[\![-]\!]$ in a CCC $\mathcal{C}$ with a functor $M : \mathcal{C}_{\mathbb{T}} \to \mathcal{C}$. More precisely, we have the following.

Lemma 4.5.3. *A* model $[\![-]\!]$ *of a λ-theory $\mathbb{T}$ in a cartesian closed category $\mathcal{C}$ determines a cartesian closed functor $M : \mathcal{C}_{\mathbb{T}} \to \mathcal{C}$ with*

$$M(\mathtt{B}) = [\![\mathtt{B}]\!], \quad M(\mathtt{c}) = [\![\mathtt{c}]\!] : 1 \to [\![C]\!] = M(C) , \tag{4.6}$$

for all basic types $\mathtt{B}$ *and basic constants* $\mathtt{c} : C$*. Moreover, M is unique up to a unique isomorphism of CCC functors, in the sense that given another model N satisfying* (4.6)*, there is a unique natural iso $M \cong N$, determined inductively by the comparison maps $M(1) \cong N(1)$,*

$$M(A \times B) \;\cong\; MA \times MB \;\cong\; NA \times NB \;\cong\; N(A \times B) ,$$

and similarly for $M(B^A)$.

Proof. Straightforward. □

We then also have the usual functorial semantics theorem:

Theorem 4.5.4. *For any λ-theory $\mathbb{T}$, the syntactic category $\mathcal{C}_{\mathbb{T}}$ classifies $\mathbb{T}$-models, in the sense that for any cartesian closed category $\mathcal{C}$ there is an equivalence of categories*

$$\mathsf{Mod}_\lambda(\mathbb{T},\mathcal{C}) \;\simeq\; \mathsf{CCC}(\mathcal{C}_{\mathbb{T}},\mathcal{C}), \tag{4.7}$$

naturally in $\mathcal{C}$. The morphisms of $\mathbb{T}$-models on the left are the isomorphisms of the underlying structures, and on the right we take the natural isomorphisms of CCC functors.

Proof. The only thing remaining to show is that, given a model $[\![-]\!]$ in a CCC $\mathcal{C}$ and a CCC functor $f : \mathcal{C} \to \mathcal{D}$, there is an induced model $[\![-]\!]^f$ in $\mathcal{D}$, given by the interpretation $[\![A]\!]^f = f[\![A]\!]$. This is straightforward, just as for algebraic theories. □

Remark 4.5.5. As mentioned in Example 4.4.5(1) the categories involved in the equivalence (4.7) are *groupoids*, in which every arrow is iso. The reason we have defined them as such is that the contravariant argument A in the function type $A \to B$ prevents us from specifying a non-iso homomorphism of models $h : M \to N$ by the obvious recursion on the type structure.

In more detail, given $h_A : [\![A]\!]^M \to [\![A]\!]^N$ and $h_B : [\![B]\!]^M \to [\![B]\!]^N$, there is no obvious candidate for a map

$$h_{A\to B} : [\![A \to B]\!]^M \longrightarrow [\![A \to B]\!]^N,$$

when all we have are the following induced maps:

$$\begin{tikzcd}
{[\![A \to B]\!]^M} \arrow[r, "="] & {([\![B]\!]^M)^{[\![A]\!]^M}} \arrow[r, "(h_B)^{[\![A]\!]^M}"] & {([\![B]\!]^N)^{[\![A]\!]^M}} & \\
& {([\![B]\!]^M)^{[\![A]\!]^N}} \arrow[u, "([\![B]\!]^M)^{h_A}"] \arrow[r, "(h_B)^{[\![A]\!]^N}"'] & {([\![B]\!]^N)^{[\![A]\!]^N}} \arrow[u, "([\![B]\!]^N)^{h_A}"'] \arrow[r, "="'] & {[\![A \to B]\!]^N}
\end{tikzcd}$$

One solution is therefore to take *isos* $h_A : [\![A]\!]^M \cong [\![A]\!]^N$ and $h_B : [\![B]\!]^M \cong [\![B]\!]^N$ and then use the inverses $h_A^{-1} : [\![A]\!]^N \to [\![A]\!]^M$ in the contravariant positions, in order to get things to line up:

$$\begin{tikzcd}
{[\![A \to B]\!]^M} \arrow[r, "="] & {([\![B]\!]^M)^{[\![A]\!]^M}} \arrow[r, "(h_B)^{[\![A]\!]^M}"] \arrow[d, "([\![B]\!]^M)^{h_A^{-1}}"', "\sim"] & {([\![B]\!]^N)^{[\![A]\!]^M}} \arrow[d, "\sim"', "([\![B]\!]^N)^{h_A^{-1}}"] & \\
& {([\![B]\!]^M)^{[\![A]\!]^N}} \arrow[r, "(h_B)^{[\![A]\!]^N}"'] & {([\![B]\!]^N)^{[\![A]\!]^N}} \arrow[r, "="'] & {[\![A \to B]\!]^N}
\end{tikzcd}$$

This suffices to at get a *category* of models $\mathsf{Mod}_\lambda\big(\mathbb{T}, \mathcal{C}\big)$, rather than just as set, which is enough structure to determine the equivalence (4.7). Note that for an algebraic theory $\mathbb{A}$, this category of λ-models in Set, say, $\mathsf{Mod}_\lambda(\mathbb{A}^\lambda)$ is still the (wide but non-full) subcategory of isomorphisms of conventional (algebraic) $\mathbb{A}$-models

$$\mathsf{Mod}_\lambda(\mathbb{A}^\lambda) \rightarrowtail \mathsf{Mod}(\mathbb{A})\,.$$

We shall consider other solutions to the problem of contravariance below.

We can now proceed just as we did in the case of algebraic theories and prove that the semantics of λ-theories in cartesian closed categories is *complete*, in virtue of the syntactic construction of the classifying category $\mathcal{C}_\mathbb{T}$. Specifically, a λ-theory $\mathbb{T}$ has a canonical interpretation $[-]$ in the syntactic category $\mathcal{C}_\mathbb{T}$, which interprets a basic type A as itself, and a basic constant c of type A as the morphism $[x : \mathtt{1} \mid c : A]$. The canonical interpretation is a model of $\mathbb{T}$, also known as the *syntactic model*, in virtue of the definition of the equivalence relation $[-]$ on terms. In fact, it is a *logically generic* model of $\mathbb{T}$, because by the construction of $\mathcal{C}_\mathbb{T}$, for any terms $\Gamma \mid u : A$ and $\Gamma \mid t : A$, we have

$$\begin{aligned} \mathbb{T} \vdash (\Gamma \mid u = t : A) \quad &\Longleftrightarrow \quad [\Gamma \mid u : A] = [\Gamma \mid t : A] \\ &\Longleftrightarrow \quad [-] \models \Gamma \mid u = t : A\,. \end{aligned}$$

For the record, we therefore have now shown:

Proposition 4.5.6. *For any λ-theory $\mathbb{T}$,*

$$\mathbb{T} \vdash (\Gamma \mid t = u : A) \quad \textit{if, and only if,} \quad [-] \models (\Gamma \mid t = u : A) \textit{ for the syntactic model } [-].$$

Of course, the syntactic model $[-]$ is the one associated under (4.7) to the identity functor $\mathcal{C}_\mathbb{T} \to \mathcal{C}_\mathbb{T}$, *i.e.* it is the *universal* one. It therefore satisfies an equation just in case the equation holds in *all* models, by the classifying property of $\mathcal{C}_\mathbb{T}$, and the preservation of satisfaction of equations by CCC functors (Proposition 4.4.2).

Corollary 4.5.7. *For any λ-theory $\mathbb{T}$,*

$$\mathbb{T} \vdash (\Gamma \mid t = u : A) \quad \textit{if, and only if,} \quad M \models (\Gamma \mid t = u : A) \textit{ for every CCC model } M.$$

Moreover, a closed type A is inhabited $\vdash a : A$ if, and only if, there is a point $1 \to [\![A]\!]^M$ in every model M.

4.6 The internal language of a CCC

In the case of algebraic theories, we were able to recover the syntactic category from the semantics by taking certain Set-valued functors on the category of models in Set. This then extended to a duality between the category of all algebraic theories and that of all "algebraic categories", which we defined as the categories of Set-valued models of some algebraic theory (and also characterized abstractly). In the (classical) propositional case, this syntax-semantics duality was seen to be exactly the classical Stone duality between the categories of Boolean algebras and of Stone topological spaces. That sort of duality theory seems to be more difficult to formulate for λ-theories, however, now that we have taken the category of models to be just a groupoid (but see Remark **??**). Nonetheless, there is still a correspondence between λ-theories and CCCs, which we get by organizing the former into a category, which is then equivalent to that of the latter. But note that this is analogous to the equivalence between algebraic theories, regarded syntactically, and regarded as finite product categories—rather than to the duality between syntax and semantics.

In order to define the equivalence in question, we first need a suitable notion of *morphism of theories*. A *translation* $\tau : \mathbb{S} \to \mathbb{T}$ of a λ-theory $\mathbb{S}$ into a λ-theory $\mathbb{T}$ is given by the following data:

1. For each basic type A in $\mathbb{S}$ a type τA in $\mathbb{T}$. The translation is then extended to all types by the rules
$$\tau \mathsf{1} = \mathsf{1}\ , \qquad \tau(A \times B) = \tau A \times \tau B\ , \qquad \tau(A \to B) = \tau A \to \tau B\ .$$
2. For each basic constant c of type A in $\mathbb{S}$ a term τc of type τA in $\mathbb{T}$. The translation of terms is then extended to all terms by the rules
$$\begin{aligned} \tau(\mathtt{fst}\, t) &= \mathtt{fst}\,(\tau t)\ , & \tau(\mathtt{snd}\, t) &= \mathtt{snd}\,(\tau t)\ , \\ \tau\langle t, u\rangle &= \langle \tau t, \tau u\rangle\ , & \tau(\lambda x : A \,.\, t) &= \lambda x : \tau A \,.\, \tau t\ , \\ \tau(t\, u) &= (\tau t)(\tau u)\ , & \tau x &= x \quad \text{(if } x \text{ is a variable)}\ . \end{aligned}$$

A context $\Gamma = x_1 : A_1, \ldots, x_n : A_n$ is translated by τ to the context
$$\tau\Gamma = x_1 : \tau A_1, \ldots, x_n : \tau A_n\ .$$

Furthermore, a translation is required to preserve the axioms of $\mathbb{S}$: if $\Gamma \mid t = u : A$ is an axiom of $\mathbb{S}$ then $\mathbb{T}$ proves $\tau\Gamma \mid \tau t = \tau u : \tau A$. It then follows that all equations proved by $\mathbb{S}$ are translated to valid equations in $\mathbb{T}$.

A moment's consideration shows that a translation $\tau : \mathbb{S} \to \mathbb{T}$ is the same thing as a model of $\mathbb{S}$ in $\mathcal{C}_{\mathbb{T}}$, despite being specified entirely syntactically. More precisely, λ-theories and translations between them clearly form a category: translations compose as functions, therefore composition is associative. The identity translation $\iota_{\mathbb{T}} : \mathbb{T} \to \mathbb{T}$ translates every type to itself and every constant to itself.

Definition 4.6.1. Let $\lambda\mathsf{Thr}$ be the category whose objects are λ-theories and morphisms are translations between them.

We now have an *isomorphism of sets*,
$$\mathsf{Hom}_{\lambda\mathsf{Thr}}(\mathbb{S}, \mathbb{T}) \cong \mathsf{Mod}_{\lambda}(\mathbb{S}, \mathcal{C}_{\mathbb{T}})\ , \tag{4.8}$$

which is natural in the theory $\mathbb{S}$, as can be seen by considering the canonical interpretation of $\mathbb{S}$ in $\mathcal{C}_{\mathbb{S}}$ induced by the identity translation $\iota_{\mathbb{S}} : \mathbb{S} \to \mathbb{S}$.

Let $\mathcal{C}$ be a small cartesian closed category. There is a λ-theory $\mathbb{L}(\mathcal{C})$ corresponding to $\mathcal{C}$, called the *internal language of* $\mathcal{C}$, and defined as follows:

1. For every object $A \in \mathcal{C}$ there is a basic type $\ulcorner A \urcorner$.
2. For every morphism $f : A \to B$ there is a basic constant $\ulcorner f \urcorner$ whose type is $\ulcorner A \urcorner \to \ulcorner B \urcorner$.
3. For every $A \in \mathcal{C}$ there is an axiom
$$x : \ulcorner A \urcorner \mid \ulcorner 1_A \urcorner x = x : \ulcorner A \urcorner \,.$$
4. For all morphisms $f : A \to B$, $g : B \to C$, and $h : A \to C$ such that $h = g \circ f$, there is an axiom
$$x : \ulcorner A \urcorner \mid \ulcorner h \urcorner x = \ulcorner g \urcorner (\ulcorner f \urcorner x) : \ulcorner C \urcorner \,.$$
5. There is a constant
$$\mathtt{T} : \mathtt{1} \to \ulcorner \mathtt{1} \urcorner \,,$$
and for all $A, B \in \mathcal{C}$ there are constants
$$\mathtt{P}_{A,B} : \ulcorner A \urcorner \times \ulcorner B \urcorner \to \ulcorner A \times B \urcorner \,, \qquad \mathtt{E}_{A,B} : (\ulcorner A \urcorner \to \ulcorner B \urcorner) \to \ulcorner B^A \urcorner \,.$$
They satisfy the following axioms:
$$\begin{aligned} u : \ulcorner \mathtt{1} \urcorner &\mid \mathtt{T} * = u : \ulcorner \mathtt{1} \urcorner \\ z : \ulcorner A \times B \urcorner &\mid \mathtt{P}_{A,B} \langle \ulcorner \pi_0 \urcorner z, \ulcorner \pi_1 \urcorner z \rangle = z : \ulcorner A \times B \urcorner \\ w : \ulcorner A \urcorner \times \ulcorner B \urcorner &\mid \langle \ulcorner \pi_0 \urcorner (\mathtt{P}_{A,B} w), \ulcorner \pi_1 \urcorner (\mathtt{P}_{A,B} w) \rangle = w : \ulcorner A \urcorner \times \ulcorner B \urcorner \\ f : \ulcorner B^A \urcorner &\mid \mathtt{E}_{A,B} (\lambda x : \ulcorner A \urcorner \,.\, (\ulcorner \mathsf{ev}_{A,B} \urcorner (\mathtt{P}_{A,B} \langle f, x \rangle))) = f : \ulcorner B^A \urcorner \\ f : \ulcorner A \urcorner \to \ulcorner B \urcorner &\mid \lambda x : \ulcorner A \urcorner \,.\, (\ulcorner \mathsf{ev}_{A,B} \urcorner (\mathtt{P}_{A,B} \langle (\mathtt{E}_{A,B} f), x \rangle)) = f : \ulcorner A \urcorner \to \ulcorner B \urcorner \end{aligned}$$

The purpose of the constants $\mathtt{T}$, $\mathtt{P}_{A,B}$, $\mathtt{E}_{A,B}$, and the axioms for them is to ensure the isomorphisms $\ulcorner \mathtt{1} \urcorner \cong \mathtt{1}$, $\ulcorner A \times B \urcorner \cong \ulcorner A \urcorner \times \ulcorner B \urcorner$, and $\ulcorner B^A \urcorner \cong \ulcorner A \urcorner \to \ulcorner B \urcorner$. Types A and B are said to be *isomorphic* if there are terms
$$x : A \mid t : B \,, \qquad y : B \mid u : A \,,$$
such that $\mathbb{S}$ proves
$$x : A \mid u[t/y] = x : A \,, \qquad y : B \mid t[u/x] = y : B \,.$$
Furthermore, an *equivalence of theories* $\mathbb{S}$ and $\mathbb{T}$ is a pair of translations
$$\begin{tikzcd} \mathbb{S} \arrow[r, bend left, "\tau"] & \mathbb{T} \arrow[l, bend left, "\sigma"] \end{tikzcd}$$

such that, for any type A in $\mathbb{S}$ and any type B in $\mathbb{T}$,

$$\sigma(\tau A) \cong A \;, \qquad\qquad \tau(\sigma B) \cong B \;.$$

The assignment $\mathcal{C} \mapsto \mathbb{L}(\mathcal{C})$ extends to a functor

$$\mathbb{L} : \mathsf{CCC} \to \lambda\mathsf{Thr} \;,$$

where CCC is the category of small cartesian closed categories and functors between them that preserve finite products and exponentials. Such functors are also called *cartesian closed functors* or *ccc functors.* If $F : \mathcal{C} \to \mathcal{D}$ is a cartesian closed functor then $\mathbb{L}(F) : \mathbb{L}(\mathcal{C}) \to \mathbb{L}(\mathcal{D})$ is the translation given by:

1. A basic type $\ulcorner A \urcorner$ is translated to $\ulcorner FA \urcorner$.
2. A basic constant $\ulcorner f \urcorner$ is translated to $\ulcorner Ff \urcorner$.
3. The basic constants $\mathtt{T}$, $\mathtt{P}_{A,B}$ and $\mathtt{E}_{A,B}$ are translated to $\mathtt{T}$, $\mathtt{P}_{FA,BA}$ and $\mathtt{E}_{FA,FB}$, respectively.

We now have a functor $\mathbb{L} : \mathsf{CCC} \to \lambda\mathsf{Thr}$. How about the other direction? We already have the construction of syntactic category which maps a λ-theory $\mathbb{S}$ to a small cartesian closed category $\mathcal{C}_{\mathbb{S}}$. This extends to a functor

$$\mathcal{C} : \lambda\mathsf{Thr} \to \mathsf{CCC} \;,$$

because a translation $\tau : \mathbb{S} \to \mathbb{T}$ induces a functor $\mathcal{C}_\tau : \mathcal{C}_{\mathbb{S}} \to \mathcal{C}_{\mathbb{T}}$ in an obvious way: a basic type $A \in \mathcal{C}_{\mathbb{S}}$ is mapped to the object $\tau A \in \mathcal{C}_{\mathbb{T}}$, and a basic constant $x : \mathtt{1} \mid c : A$ is mapped to the morphism $x : \mathtt{1} \mid \tau c : A$. The rest of $\mathcal{C}_\tau$ is defined inductively on the structure of types and terms.

Theorem 4.6.2. *The functors $\mathbb{L} : \mathsf{CCC} \to \lambda\mathsf{Thr}$ and $\mathcal{C} : \lambda\mathsf{Thr} \to \mathsf{CCC}$ constitute an equivalence of categories "up to equivalence" (a* biequivalence *of* 2-categories*). This means that for any $\mathcal{C} \in \mathsf{CCC}$ there is an equivalence of categories*

$$\mathcal{C} \simeq \mathcal{C}_{\mathbb{L}(\mathcal{C})} \;,$$

and for any $\mathbb{S} \in \lambda\mathsf{Thr}$ there is an equivalence of theories

$$\mathbb{S} \simeq \mathbb{L}(\mathcal{C}_{\mathbb{S}}) \;.$$

Proof. For a small cartesian closed category $\mathcal{C}$, consider the functor $\eta_{\mathcal{C}} : \mathcal{C} \to \mathcal{C}_{\mathbb{L}(\mathcal{C})}$, defined for an object $A \in \mathcal{C}$ and $f : A \to B$ in $\mathcal{C}$ by

$$\eta_{\mathcal{C}} A = \ulcorner A \urcorner \;, \qquad\qquad \eta_{\mathcal{C}} f = (x : \ulcorner A \urcorner \mid \ulcorner f \urcorner x : \ulcorner B \urcorner) \;.$$

To see that $\eta_{\mathcal{C}}$ is a functor, observe that $\mathbb{L}(\mathcal{C})$ proves, for all $A \in \mathcal{C}$,

$$x : \ulcorner A \urcorner \mid \ulcorner \mathtt{1}_A \urcorner x = x : \ulcorner A \urcorner$$

and for all $f : A \to B$ and $g : B \to C$,

$$x : \ulcorner A \urcorner \mid \ulcorner g \circ f \urcorner x = \ulcorner g \urcorner (\ulcorner f \urcorner x) : \ulcorner C \urcorner .$$

To see that $\eta_{\mathcal{C}}$ is an equivalence of categories, it suffices to show that for every object $X \in \mathcal{C}_{\mathbb{L}(\mathcal{C})}$ there exists an object $\theta_{\mathcal{C}} X \in \mathcal{C}$ such that $\eta_{\mathcal{C}}(\theta_{\mathcal{C}} X) \cong X$. The choice map $\theta_{\mathcal{C}}$ is defined inductively by

$$\theta_{\mathcal{C}} \mathtt{1} = \mathtt{1} , \qquad \theta_{\mathcal{C}} \ulcorner A \urcorner = A ,$$
$$\theta_{\mathcal{C}}(Y \times Z) = \theta_{\mathcal{C}} X \times \theta_{\mathcal{C}} Y , \qquad \theta_{\mathcal{C}}(Y \to Z) = (\theta_{\mathcal{C}} Z)^{\theta_{\mathcal{C}} Y} .$$

We skip the verification that $\eta_{\mathcal{C}}(\theta_{\mathcal{C}} X) \cong X$. In fact, $\theta_{\mathcal{C}}$ can be extended to a functor $\theta_{\mathcal{C}} : \mathcal{C}_{\mathbb{L}(\mathcal{C})} \to \mathcal{C}$ so that $\theta_{\mathcal{C}} \circ \eta_{\mathcal{C}} \cong 1_{\mathcal{C}}$ and $\eta_{\mathcal{C}} \circ \theta_{\mathcal{C}} \cong 1_{\mathcal{C}_{\mathbb{L}(\mathcal{C})}}$.

Given a λ-theory $\mathbb{S}$, we define a translation $\tau_{\mathbb{S}} : \mathbb{S} \to \mathbb{L}(\mathcal{C}_{\mathbb{S}})$. For a basic type A let

$$\tau_{\mathbb{S}} A = \ulcorner A \urcorner .$$

The translation $\tau_{\mathbb{S}} c$ of a basic constant c of type A is

$$\tau_{\mathbb{S}} c = \ulcorner x : \mathtt{1} \mid c : \tau_{\mathbb{S}} A \urcorner .$$

In the other direction we define a translaton $\sigma_{\mathbb{S}} : \mathbb{L}(\mathcal{C}_{\mathbb{S}}) \to \mathbb{S}$ as follows. If $\ulcorner A \urcorner$ is a basic type in $\mathbb{L}(\mathcal{C}_{\mathbb{S}})$ then

$$\sigma_{\mathbb{S}} \ulcorner A \urcorner = A ,$$

and if $\ulcorner x : A \mid t : B \urcorner$ is a basic constant of type $\ulcorner A \urcorner \to \ulcorner B \urcorner$ then

$$\sigma_{\mathbb{S}} \ulcorner x : A \mid t : B \urcorner = \lambda x : A \,.\, t .$$

The basic constants $\mathtt{T}$, $\mathtt{P}_{A,B}$ and $\mathtt{E}_{A,B}$ are translated by $\sigma_{\mathbb{S}}$ into

$$\sigma_{\mathbb{S}} \mathtt{T} = \lambda x : \mathtt{1} \,.\, x ,$$
$$\sigma_{\mathbb{S}} \mathtt{P}_{A,B} = \lambda p : A \times B \,.\, p ,$$
$$\sigma_{\mathbb{S}} \mathtt{E}_{A,B} = \lambda f : A \to B \,.\, f .$$

If A is a type in $\mathbb{S}$ then $\sigma_{\mathbb{S}}(\tau_{\mathbb{S}} A) = A$. For the other direction, we would like to show, for any type X in $\mathbb{L}(\mathcal{C}_{\mathbb{S}})$, that $\tau_{\mathbb{S}}(\sigma_{\mathbb{S}} X) \cong X$. We prove this by induction on the structure of type X:

1. If $X = \mathtt{1}$ then $\tau_{\mathbb{S}}(\sigma_{\mathbb{S}} \mathtt{1}) = \mathtt{1}$.

2. If $X = \ulcorner A \urcorner$ is a basic type then A is a type in $\mathbb{S}$. We proceed by induction on the structure of A:

   (a) If $A = \mathtt{1}$ then $\tau_{\mathbb{S}}(\sigma_{\mathbb{S}} \ulcorner \mathtt{1} \urcorner) = \mathtt{1}$. The types $\mathtt{1}$ and $\ulcorner \mathtt{1} \urcorner$ are isomorphic via the constant $\mathtt{T} : \mathtt{1} \to \ulcorner \mathtt{1} \urcorner$.

(b) If A is a basic type then $\tau_{\mathbb{S}}(\sigma_{\mathbb{S}}\ulcorner A \urcorner) = \ulcorner A \urcorner$.

(c) If $A = B \times C$ then $\tau_{\mathbb{S}}(\sigma_{\mathbb{S}}\ulcorner B \times C \urcorner) = \ulcorner B \urcorner \times \ulcorner C \urcorner$. But we know $\ulcorner B \urcorner \times \ulcorner C \urcorner \cong \ulcorner B \times C \urcorner$ via the constant $\mathsf{P}_{A,B}$.

(d) The case $A = B \to C$ is similar.

3. If $X = Y \times Z$ then $\tau_{\mathbb{S}}(\sigma_{\mathbb{S}}(Y \times Z)) = \tau_{\mathbb{S}}(\sigma_{\mathbb{S}}Y) \times \tau_{\mathbb{S}}(\sigma_{\mathbb{S}}Z)$. By induction hypothesis, $\tau_{\mathbb{S}}(\sigma_{\mathbb{S}}Y) \cong Y$ and $\tau_{\mathbb{S}}(\sigma_{\mathbb{S}}Z) \cong Z$, from which we easily obtain

$$\tau_{\mathbb{S}}(\sigma_{\mathbb{S}}Y) \times \tau_{\mathbb{S}}(\sigma_{\mathbb{S}}Z) \cong Y \times Z \ .$$

4. The case $X = Y \to Z$ is similar.

□

Composing the isomorphism 4.8 with the equivalence 4.7 we can formulate the foregoing Theorem 4.6.2 as an adjoint equivalence.

Corollary 4.6.3. *There is a biequivalence between the categories* $\lambda\mathsf{Thr}$ *of* λ*-theories and translations between them (and isos thereof), and the category* CCC *of cartesian closed categories and CCC functors (and natural isos),*

$$\begin{aligned}\mathsf{Hom}_{\lambda\mathsf{Thr}}(\mathbb{T}, \mathbb{L}\mathcal{C}) &\cong \mathsf{Mod}_{\lambda}(\mathbb{T}, \mathcal{C}) \ , \\ &\simeq \ \mathsf{Hom}_{\mathsf{CCC}}(\mathcal{C}_{\mathbb{T}}\, , \mathcal{C}) \ .\end{aligned}$$

This is mediated by an adjunction,

$$\begin{tikzcd} \mathsf{CCC} \arrow[r, bend left, "\mathbb{L}"] & \lambda\mathsf{Thr} \arrow[l, bend left, "\mathcal{C}"] \end{tikzcd}$$

with $\mathcal{C} \dashv \mathbb{L}$*, between the* syntactic category *functor* $\mathcal{C}$ *and the* internal language *functor* $\mathbb{L}$*.*

Exercise 4.6.4. In the proof of Theorem 4.6.2 we defined, for each $\mathcal{C} \in \mathsf{CCC}$, a functor $\eta_C : \mathcal{C} \to \mathcal{C}_{\mathbb{L}(\mathcal{C})}$. Verify that this determines a natural transformation $\eta : 1_{\mathsf{CCC}} \Longrightarrow \mathcal{C} \circ \mathbb{L}$ which is an equivalence of categories. What about the translation $\epsilon_{\mathbb{T}} : \mathbb{T} \to \mathbb{L}(\mathcal{C}_{\mathbb{T}})$—is that an isomorphism?

See the book [**?**] for another approach to the biequivalence of Corollary 4.6.3, which turns it into an equivalence of categories by fixing the CCC structure and requiring it to be preserved *strictly*.

4.7 Embedding and completeness theorems

We have considered the λ-calculus as a common generalization of both propositional logic, modelled by poset CCCs such as Boolean and Heyting algebras, and equational logic, modelled by finite product categories. Accordingly, there are then two different notions of "provability", as discussied in Remark 4.4.3; namely, the derivability of a closed term $\vdash a : A$, and the derivability of an equation between two (not necessarily closed) terms of the same type $\Gamma \vdash s = t : A$. With respect to the semantics, there are then two different corresponding notions of soundness and completeness: for "inhabitation" of types, and for equality of terms. We consider special cases of these notions in more detail below.

Conservativity

With regard to the former notion, inhabitation, one can also consider the question of how it compares with simple provability in *propositional logic*: e.g. a positive propositional formula ϕ in the variables $p_1, p_2, ..., p_n$ obviously determines a type Φ in the corresponding λ-theory $\mathbb{T}(X_1, X_2, ..., X_n)$ over n basic type symbols. What is the relationship between provability in positive propositional logic, $\mathsf{PPL} \vdash \phi$, and inhabitation in the associated λ-theory, $\mathbb{T}(X_1, X_2, ..., X_n) \vdash t : \Phi$? Let us call this the question of *conservativity* of λ-calculus over PPL. According to the basic idea of the Curry-Howard correspondence from Section 4.1, the λ-calculus is essentially the "proof theory of PPL". So one should expect that starting from an inhabited type Φ, a derivation of a term $\mathbb{T}(X_1, X_2, ..., X_n) \vdash t : \Phi$ should result in a corresponding proof of ϕ in PPL just by "rubbing out the proof terms". Conversely, given a provable formula $\vdash \phi$, one should be able to annotate a proof of it in PPL to obtain a derivation of a term $\mathbb{T}(X_1, X_2, ..., X_n) \vdash t : \Phi$ in the λ-calculus (although perhaps not the same term that one started with, if the proof was obtained from rubbing out a term).

We can make this idea precise semantically as follows. Write $|\mathcal{C}|$ for the poset reflection of a category $\mathcal{C}$, that is, the left adjoint to the inclusion $i : \mathsf{Pos} \hookrightarrow \mathsf{Cat}$, and let $\eta : \mathcal{C} \to |\mathcal{C}|$ be the unit of the adjunction.

Lemma 4.7.1. *If $\mathcal{C}$ is cartesian closed, then so is $|\mathcal{C}|$, and $\eta : \mathcal{C} \to |\mathcal{C}|$ preserves the CCC structure.*

Proof. Exercise! □

Exercise 4.7.2. Prove Lemma 4.7.1.

Corollary 4.7.3. *The syntactic category $\mathsf{PPC}(p_1, p_2, ..., p_n)$ of the positive propositional calculus on n propositional variables is the poset reflection of the syntactic category $\mathcal{C}_{\mathbb{T}(X_1, X_2, ..., X_n)}$ of the λ-theory $\mathbb{T}(X_1, X_2, ..., X_n)$,*

$$|\mathcal{C}_{\mathbb{T}(X_1, X_2, ..., X_n)}| \cong \mathsf{PPC}(p_1, p_2, ..., p_n)\,.$$

Proof. We already know that $\mathcal{C}_{\mathbb{T}(X_1,X_2,\dots,X_n)}$ is the free cartesian closed category on n generating objects, and that $\mathsf{PPC}(p_1,p_2,\dots,p_n)$ is the free cartesian closed poset on n generating elements. From the universal property of $\mathcal{C}_{\mathbb{T}(X_1,X_2,\dots,X_n)}$, we get a CCC map

$$\mathcal{C}_{\mathbb{T}(X_1,X_2,\dots,X_n)} \longrightarrow \mathsf{PPC}(p_1,p_2,\dots,p_n)$$

taking generators to generators, and it extends along the quotient map to $|\mathcal{C}_{\mathbb{T}(X_1,X_2,\dots,X_n)}|$ by the universal property of the poset reflection. Thus it suffices to show that the quotient map preserves, and indeed creates, the CCC structure on $|\mathcal{C}_{\mathbb{T}(X_1,X_2,\dots,X_n)}|$, which follows from the Lemma 4.7.1. □

Remark 4.7.4. Corollary 4.7.3 can be extended to other systems of type theory and logic, with further operations such as CCCs with sums $0, A+B$ ("bicartesian closed categories"), and the full intuitionistic propositional calculus IPC with the logical operations $\bot$ and $p\vee q$. We leave this as a topic for the interested student.

Completeness

As was the case for equational theories and propositional logic, the fact that there is a generic model (Proposition 4.5.6) allows the general completeness theorem stated in Corollary 4.5.7 to be specialized to various classes of special models, via embedding (or "representation") theorems, this time for CCCs, rather than for finite product categories or Boolean/Heyting algebras. We shall consider three such cases: "variable" models, Kripke models, and topological models. In each case, an "embedding theorem" of the form:

Every CCC embeds into one of the special form $\mathcal{X}$.

gives rise to a completeness theorem of the form:

For all λ-theories $\mathbb{T}$, if $1 \to [\![A]\!]^M$ in all $\mathbb{T}$-models M in all $\mathcal{X}$, then $\mathbb{T} \vdash a : A$,
and if $[\![a]\!]^M = [\![b]\!]^M : 1 \to [\![A]\!]$ in all $\mathbb{T}$-models M in all $\mathcal{X}$, then $\mathbb{T} \vdash a = b : A$.

This of course follows the same pattern that we saw for the simpler "proof relevant" case of equational (*i.e.* finite product) theories, and the even simpler "proof irrelevant" case of propositional logic, but now the proofs of some of the embedding theorems for CCCs require more sophisticated methods.

Variable models

By a *variable model* of the λ-calculus we mean one in a CCC of the form $\widehat{\mathbb{C}} = \mathsf{Set}^{\mathbb{C}^{\mathsf{op}}}$, i.e. presheaves on a (small) category $\mathbb{C}$. We regard such a model as "varying over $\mathbb{C}$", just as we saw earlier that a presheaf of groups on *e.g.* the simplex category Δ may be seen both as a simplicial group—a simplicial object in the category of groups—and as a group in the category $\mathsf{Set}^{\Delta^{\mathsf{op}}}$ of simplicial sets. The basic embedding theorem that we use in specializing Proposition 4.5.6 to such variable models is the following, which is one of the fundamental facts of categorical semantics.

Lemma 4.7.5. *For any small cartesian closed category $\mathbb{C}$, the Yoneda embedding*

$$\mathsf{y} : \mathbb{C} \hookrightarrow \mathsf{Set}^{\mathbb{C}^{\mathsf{op}}}$$

preserves the cartesian closed structure.

This is of course the "categorified" analogue of Lemma **??**, which we used for the Kripke completeness of the positive propositional calculus PPC.

Proof. We can just evaluate $\mathsf{y}A(X) = \mathbb{C}(X, A)$. It is clear that $\mathsf{y}1(X) = \mathbb{C}(X, 1) \cong 1$ naturally in X, and that $\mathsf{y}(A \times B)(X) = \mathbb{C}(X, A \times B) \cong \mathbb{C}(X, A) \times \mathbb{C}(X, B) \cong (\mathsf{y}A \times \mathsf{y}B)(X)$ for all A, B, X, naturally in all three arguments. For $B^A \in \mathbb{C}$, we then have

$$\mathsf{y}(B^A)(X) = \mathbb{C}(X, B^A) \cong \mathbb{C}(X \times A, B) \cong \widehat{\mathbb{C}}(\mathsf{y}(X \times A), \mathsf{y}B) \cong \widehat{\mathbb{C}}(\mathsf{y}X \times \mathsf{y}A, \mathsf{y}B),$$

since y is full and faithful and, as we just showed, preserves $\times$. But now recall that the exponential Q^P of presheaves P, Q is defined at X by the specification

$$Q^P(X) \;=\; \widehat{\mathbb{C}}(\mathsf{y}X \times P, Q)\,.$$

So, continuing where we left off, $\widehat{\mathbb{C}}(\mathsf{y}X \times \mathsf{y}A, \mathsf{y}B) = \mathsf{y}B^{\mathsf{y}A}(X)$, and we're done. □

For an early version of the following theorem (and much more), see the nice paper [Sco80] by Dana Scott.

Theorem 4.7.6. *For any λ-theory $\mathbb{T}$, we have the following:*

(i) A type A is inhabited,

$$\mathbb{T} \vdash a : A$$

if, and only if, for every a small category $\mathbb{C}$, in every model $[\![-]\!]$ in presheaves $\mathsf{Set}^{\mathbb{C}^{\mathsf{op}}}$ on $\mathbb{C}$, there is a point

$$1 \to [\![A]\!]\,.$$

(ii) For any terms $\Gamma \mid s, t : A$,

$$\mathbb{T} \vdash (\Gamma \mid s = t : A)$$

if, and only if,

$$[\![\Gamma \vdash s : A]\!] = [\![\Gamma \vdash t : A]\!] : [\![\Gamma]\!] \longrightarrow [\![A]\!]$$

for every such presheaf model.

Proof. We simply specialize the general completeness statement of Corollary 4.5.7 to CCCs of the form $\widehat{\mathbb{C}}$ using Lemma 4.7.5, together with the fact that the Yoneda embedding is full (and therefore reflects inhabitation) and faithful (and therefore reflects satisfaction of equations). □

4.8 Kripke models

By a Kripke model of (a theory $\mathbb{T}$ in) the λ-calculus, we mean a model $[\![-]\!]$ in the sense of Definition 4.4.1 in a presheaf CCC of the form Set^K for a poset K, *i.e.* a variable model in the sense of the previous section, where the domain of variation is just a *poset*, rather than a proper category. As with Kirpke models of propositional logic, we can regard such a model as varying through (branching) time, over a causally ordered state space, or some other partially-ordered parameter space. By Theorem 4.5.4, such a model $(K, [\![-]\!])$ is essentially the same thing as a CCC functor $M : \mathcal{C}_{\mathbb{T}} \to \mathsf{Set}^K$, taking values in "variable sets". Regarding the λ-calculus as the proof theory of the propositional calculus via the Curry-Howard correspondence (Section 4.1), it is perhaps not surprising that it should be (inhabitation) *complete* with respect to such Kripke models, in light of Theorem **??**. Completeness with respect to *equations between terms* is another matter, though; while true, the proof is far from a simple generalization of other known results. It can perhaps be seen as a verification that $\beta\eta$-equivalence is the "right" notion of equality for proofs.

Before considering such questions, however, let us first spell out explicitly what such a Kripke model looks like for the simple example of a theory $\mathbb{T}$ of an object with a commutative, binary operation,

$$\mathbb{T} \;=\; \big(\mathtt{B},\ m : \mathtt{B} \times \mathtt{B} \to \mathtt{B},\ x * y = y * x\big)\,.$$

There is one basic type symbol $\mathtt{B}$, a binary operation symbol $* : \mathtt{B} \times \mathtt{B} \to \mathtt{B}$, and a single equation $x, y : \mathtt{B} \,|\, x * y = y * x : \mathtt{B}$. Let K be a poset with ordering relation $j \leq k$ for $j, k \in K$.

A *Kripke model* M *of* $\mathbb{T}$ *over* K consists, first, of a family of sets $(M_k)_{k \in K}$, equipped with functions

$$m_{j,k} : M_j \to M_k \qquad (\text{for all } j \leq k \in K)\,,$$

satisfying the conditions:

$$m_{k,k} = 1_{M_k}, \qquad m_{j,k} \circ m_{i,j} = m_{i,k} \qquad (\text{for all } j \leq k \in K)\,.$$

This is of course exactly a functor $M : K \to \mathsf{Set}$, as the interpretation $M = [\![\mathtt{B}]\!]$ of the basic type symbol $\mathtt{B}$. Next, we need functions

$$s_k : M_k \times M_k \to M_k \qquad (\text{for all } k \in K)$$

satisfying

$$s_k\big(m_{j,k}(x), m_{j,k}(y)\big) \;=\; m_{j,k}\big(s_j(x, y)\big) \qquad (\text{for all } j \leq k \in K \text{ and } x, y \in M_j)\,.$$

This is just a natural transformation $s : M \times M \to M$, as the interpretation $s = [\![*]\!]$ of the operation symbol $* : \mathtt{B} \times \mathtt{B} \to \mathtt{B}$. Finally, the interpretation $(M, s) = [\![\mathtt{B}, *]\!]$ should satisfy the equation $x, y : \mathtt{B} \,|\, x * y = y * x : \mathtt{B}$, meaning that

$$s_k(x, y) = s_k(y, x) \qquad (\text{for all } k \in K)\,,$$

since two natural transformations are equal just if all of their components are equal. Thus a Kripke model of this theory $\mathbb{T}$ is just a model of the underlying algebraic theory in the functor category Set^K—which of course is the same thing as a functor from K to the category of $\mathbb{T}$-models in Set.

A theory involving an operation of "higher type", such as the section $s : (D \to D) \to D$ in (the theory of) a reflexive type (Example 4.3.7) is no more "non-standard". Let $D = [\![\mathtt{D}]\!]$ be the interpretation of the basic type $\mathtt{D}$, so that $[\![\mathtt{D} \to \mathtt{D}]\!] = D^D : K \to \mathsf{Set}$ is an exponential presheaf. At each $k \in K$, we then have,

$$(D^D)_k = \mathsf{Set}^K\big(D \times K(k, -), D\big)\,,$$

which is trivial except on the upset $\uparrow k$, where it consists of natural transformations

$$\mathsf{Set}^{\uparrow k}\big(D{\uparrow}k, D{\uparrow}k\big)\,,$$

where $D{\uparrow}k : \uparrow k \to \mathsf{Set}$ is just D restricted to the upset $\uparrow k \subseteq K$, *i.e.* the composite

$$\uparrow k \hookrightarrow K \xrightarrow{D} \mathsf{Set}\,.$$

Given any such natural transformation $\vartheta : D{\uparrow}k \longrightarrow D{\uparrow}k$, and any $k \leq j$, the action of the functor,

$$(D^D)_k \to (D^D)_j$$

on ϑ is simply to restrict it further to $\uparrow j \subseteq \uparrow k$, thus taking ϑ to

$$\vartheta{\uparrow}j : D{\uparrow}j \longrightarrow D{\uparrow}j\,,$$

which is just the same function as ϑ, with the new domain of definition $\uparrow j \subseteq \uparrow k$.

The section $s : (D \to D) \to D$ therefore takes, at each $k \in K$, such a $\vartheta : D{\uparrow}k \longrightarrow D{\uparrow}k$ to an element $s_k(\vartheta) \in D_k$, respecting the restrictions $\uparrow j \subseteq \uparrow k$ in the sense that

$$d_{k,j}s_k(\vartheta) = s_j(\vartheta{\uparrow}j) \in D_j\,,$$

where $d_{k,j} : D_k \to D_j$ is the action of the functor $D : K \to \mathsf{Set}$.

In this way, the presheaf exponential $D^D : K \to \mathsf{Set}$ is entirely determined by the "base-case" $D : K \to \mathsf{Set}$, and is still a "full function space" at each $k \in K$, but the functorial action in k requires it not to be just $D_k^{D_k}$ (which for a reflexive type would then be trivial at all $k \in K$), but rather, to take the entire segment $\uparrow k$ into account—much in the way that $k \Vdash \varphi \Rightarrow \psi$ was determined for Kripke models of the intuitionistic propositional calculus IPC by considering all $j \geq k$. (Indeed, one can explicitly formulate the Kripke semantics for simple type theory in the usual Kripke-forcing style $k \Vdash a : A$, *cf.* [AGH21].)

The proof of the following theorem uses a deep result from topos theory (due to Joyal-Tierney [**?**]) that is, unfortunately, beyond the scope of this book. It implies that, for every small CCC $\mathcal{C}$ there is a poset K and a full and faithful CCC functor $\mathcal{C} \hookrightarrow \mathsf{Set}^K$.

Theorem 4.8.1 (Kripke completeness for λ-calculus). *For any λ-theory $\mathbb{T}$:*

(i) A type A is inhabited just if it has a point $1 \to [\![A]\!]$ in every Kripke model $(K, [\![-]\!])$.

(ii) Two terms are provably equal, $\mathbb{T} \vdash (\Gamma \mid s = t : A)$, just if they are equal in every Kripke model $(K, [\![-]\!])$,

$$[\![s]\!] = [\![t]\!] : [\![\Gamma]\!] \longrightarrow [\![A]\!]\,.$$

For the proof, see [AR11], as well as [AGH21].

Remark 4.8.2. One can reformulate the Kripke semantics for simple type theory in terms of *discrete opfibrations* of posets,

$$\pi : F \longrightarrow K\,,$$

rather than (covariant) presheaves $F : K \to \mathsf{Set}$. Indeed, since the (full!) subcategory of all such maps

$$\mathsf{dopFib}/_K \hookrightarrow \mathsf{Pos}/_K$$

is equivalent to Set^K, this category is also cartesian closed. And with its obvious forgetful functor

$$\mathsf{dopFib}/_K \longrightarrow \mathsf{Pos}\,,$$

this provides another useful perspective on the functor category Set^K. This "fibrational" point of view is pursued in [AR11]. It is particularly useful for the semantics of *dependent* type theory, which we shall consider in next.

4.9 Dependent type theory

The Curry-Howard correspondence from Section 4.1 can be extended to natural deduction proofs in *first-order* logic, providing a refinement of the "propositions as types/proofs as terms" idea from propositional to first-order logic. In addition to simple types $A, B, ...$ representing propositions, one has *dependent* types $x : A \mid B(x)$ representing propositional functions. In addition to the simple type formers $A \times B$ and $A \to B$, one has dependent type formers $\Sigma_{x:A}B(x)$ and $\Pi_{x:A}B(x)$, representing the quantified formulas $\exists_{x:A}B(x)$ and $\forall_{x:A}B(x)$. As before, these types may have different terms $s, t : \Pi_{x:A}B(x)$, resulting from different proofs of the corresponding propositions, so that the calculus of terms records more information than mere *provability.* Also as before, the resulting structure turns out to be one that is shared by other categories *not* arising from logic—and now the coincidence is even more remarkable, because the structure at issue is a much richer and more elaborate one. Where proofs in the propositional calculus gave rise to a Cartesian closed category, the category of proof terms of first-order logic will be seen to be *locally Cartesian closed*, a mathematical structure also shared by sheaves on a space, Grothendieck toposes, categories of fibrations, and other important examples.

Recall first the notion of a hyperdoctrine $P : \mathcal{C}^{\mathsf{op}} \to \mathsf{Cat}$ from Section **??**, and in particular the distinction between poset-valued and proper ones. The latter correspond more closely to dependent type theory, where the individual value categories $P(C)$ may be, e.g., cartesian closed, but they must also admit adjoints $\Sigma_A \dashv p_A^* \dashv \Pi_A$ along all projections $p_A : X \times A \to A$ in the category $\mathcal{C}$ of contexts. An important *difference* between hyperdoctrines and dependent type theories, however, is that the category of contexts in dependent type theory has not just finite products or finite limits, but also additional structure resulting from an operation of *context extension*, which takes as input a type in context $\Gamma \mid A$ and returns a new context $(\Gamma, x : A)$ together with a substitution $(\Gamma, x : A) \to \Gamma$. This is taking the "propositions-as-types" idea seriously, by allowing every proposition $\Gamma \mid \varphi$ in first-order logic to form a new type $\{\Gamma \mid \varphi\}$, or letting the objects $A \in P(C)$ in a hyperdoctrine $(\mathcal{C}, P)$ become arrows $\{A\} \to C$ in $\mathcal{C}$.[2]

Dependently-typed lambda-calculus. We give a somewhat informal specification of the syntax of the *dependently-typed λ-calculus* (see [**?**] for a more detailed exposition). To formulate the rules, we revisit the rules of simple type theory from section 4.3 and adjust them as follows.

Judgements: There are three kinds of judgements: for contexts, types, and terms, respectively,

$$\Gamma\ \mathtt{ctx}\ , \qquad \Gamma \mid A\ \mathtt{type}\ , \qquad \Gamma \mid a : A\,.$$

For each of these there are also (judgemental) equalities, the rules for which are the expected ones.

Contexts: These are formed by the rules:

$$\frac{}{(\cdot)\ \mathtt{ctx}} \qquad\qquad \frac{\Gamma \mid A\ \mathtt{type}}{\Gamma, x : A\ \mathtt{ctx}}$$

Here it is assumed that x is a fresh variable, not already occurring in Γ. Note that the order of the types occurring in a context matters, since types to the right may depend on ones to their left.

Types: In addition to the usual *simple types*, generated from *basic types* by formation of *products* and *function types*, we may also have some *basic types in context*,

$$\text{Basic dependent types} \quad \Gamma_1 \mid \mathtt{B}_1,\ \Gamma_2 \mid \mathtt{B}_2,\ \cdots$$

where the contexts Γ need not be basic. Further dependent types are formed from the basic ones by the Σ and Π type formers, using the rules:

$$\frac{\Gamma, x : A \mid B\ \mathtt{type}}{\Gamma \mid \Sigma_{x:A} B\ \mathtt{type}} \qquad\qquad \frac{\Gamma, x : A \mid B\ \mathtt{type}}{\Gamma \mid \Pi_{x:A} B\ \mathtt{type}}$$

[2] [Law70] does just this.

Terms: As for simple types, we assume there is a countable set of *variables* $x, y, z, \ldots$. We are also given a set of *basic constants*. The set of *terms* is generated from variables and basic constants by the following grammar, just as for simple types:

$$\begin{aligned} \text{Variables} \quad & v ::= x \mid y \mid z \mid \cdots \\ \text{Constants} \quad & c ::= \mathtt{c}_1 \mid \mathtt{c}_2 \mid \cdots \\ \text{Terms} \quad & t ::= v \mid c \mid * \mid \langle t_1, t_2 \rangle \mid \mathtt{fst}\, t \mid \mathtt{snd}\, t \mid t_1\, t_2 \mid \lambda x : A \,.\, t \end{aligned}$$

The rules for deriving typing judgments are as for simple types:

- Each basic constant $\mathtt{c}_i$ has a uniquely determined type C_i (not necessarily basic):

$$\frac{}{\Gamma \mid \mathtt{c}_i : C_i}$$

- The type of a variable is determined by the context:

$$\frac{}{x_1 : A_1, \ldots, x_i : A_i, \ldots, x_n : A_n \mid x_i : A_i} \ (1 \leq i \leq n)$$

- The constant $*$ has type $\mathtt{1}$:

$$\frac{}{\Gamma \mid * : \mathtt{1}}$$

- The typing rules for pairs and projections now take the form:

$$\frac{\Gamma \mid a : A \qquad \Gamma \mid b : B(a)}{\Gamma \mid \langle a, b \rangle : \Sigma_{x:A} B} \qquad \frac{\Gamma \mid c : \Sigma_{x:A} B}{\Gamma \mid \mathtt{fst}\, c : A} \qquad \frac{\Gamma \mid c : \Sigma_{x:A} B}{\Gamma \mid \mathtt{snd}\, c : B(\mathtt{fst}\, c)}$$

  We write e.g. $B(a)$ rather than $B[a/x]$ to indicate a substitution of the term a for the variable x in the type B. We treat $A \times B$ as another way of writing $\Sigma_{x:A} B$, when the variable $x : A$ does not occur in the type B.

- The typing rules for application and λ-abstraction are now:

$$\frac{\Gamma \mid t : \Pi_{x:A} B \qquad \Gamma \mid a : A}{\Gamma \mid t\, a : B(a)} \qquad \frac{\Gamma, x : A \mid t : B}{\Gamma \mid (\lambda x : A \,.\, t) : \Pi_{x:A} B}$$

  We treat $A \to B$ as another way of writing $\Pi_{x:A} B$, when the variable $x : A$ does not occur in the type B.

Equations: The (β and η) equations between these terms are just as they were for simple types. There are also the usual equations making judgemental equality a congruence with respect to all type and term formers.

Equality types: Just as for first-order logic, we may also add a primitive *equality type* $x =_A y$ for each type A, sometimes called *propositional equality*, and not to be confused with the judgemental equality, which we shall now write as $s \equiv t$ to emphasize the difference. The formation, introduction, elimination, and computation rules for equality types are as follows:

$$\frac{\Gamma \mid s : A \qquad \Gamma \mid t : A}{\Gamma \mid s =_A t \ \texttt{type}} \qquad\qquad \frac{\Gamma \mid a : A}{\Gamma \mid \texttt{refl}_a : (a =_A a)}$$

$$\frac{\Gamma \mid p : s =_A t}{\Gamma \mid s \equiv t : A} \qquad\qquad \frac{\Gamma \mid p : s =_A t}{\Gamma \mid p \equiv \texttt{refl}_s : (s =_A s)}$$

Remark 4.9.1 (Identity types)**.** This formulation of the rules for equality is known as the *extensional* theory. There is also an *intensional* version, with different elimination (and computation) rules, in which the types are sometimes called *identity types* and written $\texttt{Id}_A(s,t)$ instead. See [**?**] for details.

Example 4.9.2 (The type-theoretic axiom of choice)**.** Reading Σ as "there exists" and Π as "for all", a type such a $\Pi_{x:A}\Sigma_{y:B}R(x,y)$ can be regarded as a stating a proposition—in this case, "for all $x : A$ there is a $y : B$ such that $R(x,y)$". By Curry-Howard, such a "proposition" is then provable if it has a closed term $t : \Pi_{x:A}\Sigma_{y:B}R(x,y)$, which then corresponds to a proof, by unwinding the rules that constructed the term, and observing that they correspond to the usual natural deduction rules for first-order logic.

This only partly true, however: the rules of construction for terms correspond to provability under a certain "constructive" conception of validity (see [**?**]). This is made clear by the following example, which is sometimes called the "type theoretic axiom of choice", because it sounds like the axiom of choice under the conventional interpretation; but this statement is actually provable from the rules of type theory, rather than being an axiom!

$$\Pi_{x:A}\Sigma_{y:B}R(x,y) \to \Sigma_{f:A\to B}\Pi_{x:A}R(x,fx)\,. \tag{4.9}$$

Exercise 4.9.3. Prove the type theoretic axiom of choice (4.9) from the rules for dependent type theory given here.

4.10 Locally cartesian closed categories

Recall the following from Proposition **??**.

Proposition 4.10.1. *The following conditions on a category $\mathcal{C}$ with terminal object* 1 *are equivalent:*

1. *Every slice category $\mathcal{C}/A$ is cartesian closed.*

2. *For every arrow $f : B \to A$, the (post-) composition functor $\Sigma_f : \mathcal{C}/B \to \mathcal{C}/A$ has a right adjoint f^*, which in turn has a right adjoint Π_f,*

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

$$\begin{tikzcd} \mathcal{C}/B \arrow[r, bend left, "\Sigma_f"] \arrow[r, bend right, "\Pi_f"'] & \mathcal{C}/A \arrow[l, "f^*" description] \end{tikzcd}$$

Such a category is called locally cartesian closed*.*

The notation of course anticipates the interpretation of DTT.

Proof. Construct Π from exponentials and pullbacks; see the proof of Proposition **??**. $\square$

Basic examples of LCCCs

We have the following basic examples, most of which we have already seen in Section **??** on hyperdoctrines,

1. Set: We have already seen the hyperdoctrine Set^I of *families of sets* $(A_i)_{i\in I}$, with action of $f : J \to I$ on $A : I \to \mathsf{Set}$ by precomposition $f^*A = A \circ f : J \to \mathsf{Set}$. The equivalent hyperdoctrine
$$\mathsf{Set}^I \ \simeq\ \mathsf{Set}/_I$$
uses the slice categories $\mathsf{Set}/_I$ with action by pullback $f^* : \mathsf{Set}/_I \to \mathsf{Set}/_J$. It follows that Set is locally cartesian closed.

2. Presheaves: The LCC structure on presheaves $\widehat{\mathbb{C}}$ on a small category $\mathbb{C}$ follows from the CCC structure on each slice, since each of the slice categories $\widehat{\mathbb{C}}/_X$ is another category of presheaves, namely $\widehat{\int_{\mathbb{C}} X}$, on the category of elements $\int_{\mathbb{C}} X$. That $\widehat{\mathbb{C}}$ is a CCC is shown directly by computing the products of presheaves $P \times Q$ pointwise, and the exponential as $Q^P = \mathsf{Hom}(\mathsf{y}(-) \times P, Q)$.

3. Pos: The category of posets is cartesian closed, but not locally so. However, we have seen that the category of discrete fibartions on a poset K is equivalent to a category of presheaves $\mathsf{dFib}(K) \simeq \mathsf{Set}^{K^{\mathsf{op}}}$. It follows that the (non-full) subcategory $\mathsf{dFib} \hookrightarrow \mathsf{Pos}$ of posets and discrete fibrations as arrows would be locally cartesian closed *except* for the fact that it lacks a terminal object. Thus every slice of this category $\mathsf{dFib}(K)/P \simeq \mathsf{Set}^{K^{\mathsf{op}}}/P \simeq \mathsf{Set}^{(\int_K P)^{\mathsf{op}}}$ *is* LCC.

4. An example similar to the foregoing is the non-full subcategory $\mathsf{LocHom} \hookrightarrow \mathsf{Top}$ of topological spaces and local homeomorphisms between them, which lacks a terminal object, but each slice of which $\mathsf{LocHom}/X \simeq \mathsf{Sh}(X)$ is equivalent to the topos of *sheaves* on the space X, and is therefore CCC (and so LCCC).

Exercise 4.10.2. Let $P : \mathcal{C}^{\mathsf{op}} \to \mathsf{Cat}$ be a hyperdoctrine for which there are equivalences $PC \simeq \mathcal{C}/C$, naturally in C, with respect to the left adjoints $\Sigma_f : \mathcal{C}/A \to \mathcal{C}/B$ for all $f : A \to B$ in $\mathcal{C}$. Show that $\mathcal{C}$ is then LCC.

Exercise 4.10.3. Show that any LCCC $\mathcal{C}$, regarded as a hyperdoctrine, has equality in the sense of Remark **??**.

4.11 Functorial semantics of DTT in LCCCs

In the semantics of dependent type theory in a locally cartesian closed category, contexts are interpreted as objects, and dependent types as morphisms. Let $\mathcal{C}$ be an LCCC and interpret the empty context as the terminal object, $[\![\cdot]\!] = 1$, and for a closed type $\cdot \mid B$, let $[\![\cdot \mid B]\!] : [\![B]\!] \to 1$. More generally, given any type in context $\Gamma \mid A$, we shall have

$$[\![\Gamma \mid A]\!] : [\![\Gamma, A]\!] \longrightarrow [\![\Gamma]\!] ,$$

abbreviating $\Gamma, x : A$ to Γ, A. Specifically, given $\Gamma, A \mid B$, we then have maps

$$[\![\Gamma, A, B]\!] \xrightarrow{[\![\Gamma, A \mid B]\!]} [\![\Gamma, A]\!] \xrightarrow{[\![\Gamma \mid A]\!]} [\![\Gamma]\!] ,$$

and we use the left and right adjoints to pullback to interpret the eponymous type-forming operations:

$$[\![\Gamma \mid \Sigma_{x:A} B]\!] = \Sigma_{[\![\Gamma \mid A]\!]}([\![\Gamma, A \mid B]\!]) ,$$
$$[\![\Gamma \mid \Pi_{x:A} B]\!] = \Pi_{[\![\Gamma \mid A]\!]}([\![\Gamma, A \mid B]\!]) .$$

A term $\Gamma \mid a : A$ is interpreted as a section:

$$\begin{tikzcd}
{[\![\Gamma]\!]} \arrow[r, "{[\![\Gamma \mid a : A]\!]}"] \arrow[dr, "="'] & {[\![\Gamma, A]\!]} \arrow[d, "{[\![\Gamma \mid A]\!]}"] \\
 & {[\![\Gamma]\!]}
\end{tikzcd}$$

Finally, as in first-order logic, substitution of a term $\Gamma \mid a : A$ for a variable $\Gamma, x : A$ is interpreted by pullback,

$$\begin{tikzcd}
{[\![\Gamma, B(a)]\!]} \arrow[r] \arrow[d, "{[\![\Gamma \mid B(a)]\!]}"'] \arrow[dr, phantom, "\lrcorner", very near start] & {[\![\Gamma, A, B]\!]} \arrow[d, "{[\![\Gamma, A \mid B]\!]}"] \\
{[\![\Gamma]\!]} \arrow[r, "{[\![\Gamma \mid a : A]\!]}"'] & {[\![\Gamma, A]\!]}
\end{tikzcd}$$

and similarly for substitution into terms. The interpretation of substitution as pullback leads to a coherence problem that we shall consider in the next section.

As was done for simple type theory in Section 4.6, we can again develop the relationship between the type theory and its models using the framework of *functorial semantics*. This is now a common generalization of λ-theories, modeled in CCCs, and first-order logic, modeled in Heyting categories. The first step is to build a syntactic *classifying category* $\mathcal{C}_{\mathbb{T}}$ from a theory $\mathbb{T}$ in dependent type theory, which we then show classifies $\mathbb{T}$-models in LCCCs. We omit the now essentially routine details (given the analogous cases already considered), and merely state the main result, the proof of which is also analogous to the previous cases. A detailed treatment can be found in the seminal paper [See84].

Theorem 4.11.1. *For any theory $\mathbb{T}$ in dependent type theory, the locally cartesian closed syntactic category $\mathcal{C}_{\mathbb{T}}$ classifies $\mathbb{T}$-models, in the sense that for any locally cartesian closed category $\mathcal{C}$ there is an equivalence of categories*

$$\mathsf{Mod}(\mathbb{T},\mathcal{C})^i \simeq \mathsf{LCCC}(\mathcal{C}_{\mathbb{T}},\mathcal{C})^i, \tag{4.10}$$

naturally in $\mathcal{C}$. The morphisms of $\mathbb{T}$-models on the left are the isomorphisms of the underlying structures, and on the right we take the natural isomorphisms of LCCC functors.

As a corollary, again as before, we have that dependent type theory is *complete* with respect to the semantics in locally cartesian closed categories, in virtue of the syntactic construction of the classifying category $\mathcal{C}_{\mathbb{T}}$. Specifically, any theory $\mathbb{T}$ has a canonical interpretation $[-]$ in the syntactic category $\mathcal{C}_{\mathbb{T}}$ which is *logically generic* in the sense that, for any terms $\Gamma \mid s : A$ and $\Gamma \mid t : A$, we have

$$\begin{aligned} \mathbb{T} \vdash (\Gamma \mid u \equiv t : A) \quad &\iff \quad [\Gamma \mid u : A] = [\Gamma \mid t : A] \\ &\iff \quad [-] \models (\Gamma \mid s \equiv t : A)\,. \end{aligned}$$

Thus, for the record, we have:

Proposition 4.11.2. *For any dependently typed theory $\mathbb{T}$,*

$$\mathbb{T} \vdash (\Gamma \mid u \equiv t : A) \quad \textit{if, and only if,} \quad \mathcal{C}_{\mathbb{T}} \models (\Gamma \mid u \equiv t : A)\,.$$

Of course, the syntactic model $[-]$ in $\mathcal{C}_{\mathbb{T}}$ is the one associated under (4.10) to the identity functor $\mathcal{C}_{\mathbb{T}} \to \mathcal{C}_{\mathbb{T}}$, *i.e.* it is the *universal* one. It therefore satisfies an equation just in case the equation holds in *all* models, by the classifying property of $\mathcal{C}_{\mathbb{T}}$, and the preservation of satisfaction of equations by LCCC functors (as in Proposition 4.4.2).

Corollary 4.11.3. *For any dependently typed theory $\mathbb{T}$,*

$$\mathbb{T} \vdash (\Gamma \mid u \equiv t : A) \quad \textit{if, and only if,} \quad M \models (\Gamma \mid u \equiv t : A) \textit{ for every LCCC model } M.$$

Moreover, a closed type A is inhabited $\vdash a : A$ if, and only if, there is a point $1 \to [\![A]\!]^M$ in every model M.

The embedding and completeness theorems of Section 4.7 with respect to general presheaf and Kripke models can also be extended to dependently typed theories. See [AR11] for details. There is also a version of Kripke-Joyal forcing for such theories (and an associated completeness theorem), for which the interested reader can consult [AGH21].

4.12 Coherence and natural models

The semantics of DTT in LCCCs described in the previous section uses the "slice category" hyperdoctrine of an LCC to interpret the dependent types. Thus the contexts Γ and substitutions $\sigma : \Delta \to \Gamma$ are interpreted as the objects and arrow of an LCC category $\mathcal{C}$, and the dependent types $\Gamma \mid A$ and terms $\Gamma \mid a : A$ are interpreted as objects $A \to \Gamma$ in the slice category $\mathcal{C}/\Gamma$ and their global sections $a : \Gamma \to A$ (over Γ). However, there is a problem with this kind of semantics (as first pointed out by [Hof]): as a hyperdoctrine, this interpretation is a *pseudo*functor $\mathcal{C}/ : \mathcal{C}^{\mathsf{op}} \to \mathsf{Cat}$, but the syntax of DTT produces an actual *presheaf* of types in context $\mathsf{Ty} : \mathcal{C}^{\mathsf{op}} \to \mathsf{Set}$, since substitution into dependent types is *strictly* functorial with respect to composition of substitutions, in the sense that for a type in context $\Gamma \mid A$ and substitutions $\sigma : \Delta \to \Gamma$ and $\tau : \Theta \to \Delta$ we have an *equality of types in context*,

$$\Theta \mid (A[\sigma])[\tau] \equiv A[\sigma \circ \tau]\,,$$

rather than the (canonical) isomorphism $\cong$ fitting into the two-pullbacks diagram of the hyperdoctrine, namely:

$$\begin{tikzcd}
(\sigma\circ\tau)^*A \arrow[r, "\cong"'] \arrow[dr] & \tau^*\sigma^*A \arrow[r] \arrow[d] \arrow[dr, phantom, "\lrcorner", very near start] & \sigma^*A \arrow[r] \arrow[d] \arrow[dr, phantom, "\lrcorner", very near start] & A \arrow[d] \\
& \Theta \arrow[r, "\tau"] \arrow[rr, bend right, "\sigma\circ\tau"'] & \Delta \arrow[r, "\sigma"] & \Gamma
\end{tikzcd}$$

with an additional arrow $\sigma^*A \to A$ drawn bending over:

A similar problem occurs in the Beck-Chavalley conditions, where the hyperdoctrine structure has only canonical isos, rather than the strict equalities that obtain in the syntax, such as

$$(\Pi_{x:A}B)[\sigma] \equiv (\Pi_{x:A[\sigma]}B[\sigma])\,.$$

There are various different solutions to this problem in the literature, some involving "strictifications" of the LCC slice-category hyperdoctrine (including both left- and right-adjoint strictifications), as well as other semantics altogether, such as categories-with-families [Dyb96], categories-with-attributes, and comprehension categories.

A solution based on the notion of *universe* $\widetilde{U} \to U$ was first proposed by Voevodsky; this approach is combined with the notion of a *representable natural transformation* in [Awo16] as follows.

Definition 4.12.1. For a small category $\mathbb{C}$, a natural transformation $f : Y \to X$ of presheaves on $\mathbb{C}$ is called *representable* if for every $C \in \mathbb{C}$ and $x \in X(C)$, there is given a

$D \in \mathbb{C}$, a $p : D \to C$, and a $y \in Y(D)$ such that the following square is a pullback.

$$\begin{tikzcd}
\mathsf{y}D \arrow[r, "y"] \arrow[d, "\mathsf{y}p"'] \arrow[dr, phantom, "\lrcorner", very near start] & Y \arrow[d, "f"] \\
\mathsf{y}C \arrow[r, "x"'] & X
\end{tikzcd} \tag{4.11}$$

A representable natural transformation is the same thing as a *category with families* in the sense of Dybjer [Dyb96]. Indeed, let us write the objects of $\mathbb{C}$ as $\Gamma, \Delta, \ldots$ and the arrows as $\sigma : \Delta \to \Gamma, \ldots$, thinking of $\mathbb{C}$ as a "category of contexts". Let $p : E \to U$ be a representable map of presheaves, and write its elements as:

$$\begin{aligned} A \in U(\Gamma) &\quad \text{iff} \quad \Gamma \mid A \\ a \in E(\Gamma) &\quad \text{iff} \quad \Gamma \mid a : A, \end{aligned}$$

where $A = p \circ a$, as indicated in:

$$\begin{tikzcd}
 & E \arrow[d, "p"] \\
\mathsf{y}\Gamma \arrow[ur, "a"] \arrow[r, "A"'] & U.
\end{tikzcd}$$

Thus we regard U as the *presheaf of types*, with $U(\Gamma)$ the set of all types in context Γ, and E as the *presheaf of terms*, with $E(\Gamma)$ the set of all terms in context Γ, while the component $p_\Gamma : E(\Gamma) \to U(\Gamma)$ is the typing of the terms in context Γ.

Naturality of $p : E \to U$ just means that for any substitution $\sigma : \Delta \to \Gamma$, we have an action on types and terms:

$$\begin{aligned} \Gamma \mid A &\quad \mapsto \quad \Delta \mid A\sigma \\ \Gamma \mid a : A &\quad \mapsto \quad \Delta \mid a\sigma : A\sigma\,. \end{aligned}$$

While, by functoriality, given any further $\tau : \Theta \to \Delta$, we have

$$(A\sigma)\tau = A(\sigma \circ \tau) \qquad (a\sigma)\tau = a(\sigma \circ \tau),$$

as well as

$$A1 = A \qquad a1 = a$$

for the identity substitution $1 : \Gamma \to \Gamma$.

Finally, the representability of the natural transformation $p : E \to U$ is exactly the operation of *context extension*: given any $\Gamma \mid A$, by Yoneda we have the corresponding map $A : \mathsf{y}\Gamma \to U$, and we let $p_A : \Gamma.A \to \Gamma$ be (the map representing) the pullback of p along A, as in (4.11). We therefore have a pullback square:

$$\begin{tikzcd}
\mathsf{y}\Gamma.A \arrow[r, "q_A"] \arrow[d, "\mathsf{y}p_A"'] \arrow[dr, phantom, "\lrcorner", very near start] & E \arrow[d, "p"] \\
\mathsf{y}\Gamma \arrow[r, "A"'] & U,
\end{tikzcd} \tag{4.12}$$

where the map $q_A : \Gamma.A \to E$ now determines a term

$$\Gamma.A \mid q_A : Ap_A.$$

We may omit the y for the Yoneda embedding, letting the Greek letters serve to distinguish representable presheaves.

Exercise 4.12.2. Show that the fact that (4.12) is a pullback means that given any $\sigma : \Delta \to \Gamma$ and $\Delta \mid a : A\sigma$, there is a map

$$(\sigma, a) : \Delta \to \Gamma.A,$$

and this operation satisfies the equations

$$\begin{aligned} p_A \circ (\sigma, a) &= \sigma \\ q_A(\sigma, a) &= a, \end{aligned}$$

as indicated in the following diagram.

$$\begin{tikzcd}
\Delta \arrow[dr, dotted, "{(\sigma, a)}"] \arrow[drr, bend left, "a"] \arrow[ddr, bend right, "\sigma"'] & & \\
& \Gamma.A \arrow[r, "q_A"'] \arrow[d, "p_A"] \arrow[dr, phantom, "\lrcorner", very near start] & E \arrow[d, "p"] \\
& \Gamma \arrow[r, "A"'] & U
\end{tikzcd}$$

Show moreover that the uniqueness of (σ, a) means that for any $\tau : \Delta' \to \Delta$ we also have:

$$\begin{aligned} (\sigma, a) \circ \tau &= (\sigma \circ \tau, a\tau) \\ (p_A, q_A) &= 1. \end{aligned}$$

Comparing the foregoing with the definition of a category with families in [Dyb96], we have shown:

Proposition 4.12.3. *Let $p : E \to U$ be a natural transformation of presheaves on a small category $\mathbb{C}$ with a terminal object. Then p is representable in the sense of Definition 4.12.1 just in case it determines a category with families, as just indicated.*

The notion of a category with families is a variable-free way of presenting dependent type theory, including contexts and substitutions, types and terms in context, and context extension. Accordingly, we may think of a representable map of presheaves on a category $\mathbb{C}$ as a "type theory over $\mathbb{C}$" as the category of contexts and substitutions. (This is the reason for the requirement that $\mathbb{C}$ should have a terminal object to represent the "empty context".) One can also show that such a map of presheaves is essentially determined by a class of maps in $\mathbb{C}$ that is closed under all pullbacks, corresponding to the types in context (see [Awo16]).

Definition 4.12.4. A *natural model of type theory* on a small category $\mathbb{C}$ is a representable map of presheaves $p : E \to U$.

Exercise 4.12.5. Let $\mathbb{T}$ be a dependent type theory and $\mathcal{C}_{\mathbb{T}}$ its category of contexts and substitutions. Define the presheaves $\mathsf{Ty} : \mathcal{C}_{\mathbb{T}}{}^{\mathsf{op}} \to \mathsf{Set}$ of types-in-context and $\mathsf{Tm} : \mathcal{C}_{\mathbb{T}}{}^{\mathsf{op}} \to \mathsf{Set}$ of terms-in-context, along with a natural transformation,

$$\tau : \mathsf{Tm} \to \mathsf{Ty}$$

that takes a term to its type. Show that $\tau : \mathsf{Tm} \to \mathsf{Ty}$ is a natural model of type theory.

4.13 Universes

4.14 Induction and W-types

Bibliography

[AF13] S. Awodey and H. Forssell. First-order logical duality. *Annals of Pure and Applied Logic*, 164(3):319–348, 2013.

[AGH21] S. Awodey, N. Gambino, and S. Hazratpour. Kripke-Joyal forcing for type theory and uniform fibrations, October 2021. Preprint available as `https://arxiv.org/abs/2110.14576`.

[AR11] S. Awodey and F. Rabe. Kripke semantics for Martin-Löf's extensional type theory. *Logical Methods in Computer Science*, 7(3):1–25, 2011.

[Awo10] Steve Awodey. *Category Theory.* Number 52 in Oxford Logic Guides. Oxford University Press, 2010.

[Awo16] Steve Awodey. Natural models of homotopy type theory. *Mathematical Structures in Computer Science*, 28:1–46, 11 2016.

[Awo21] Steve Awodey. Sheaf representations and duality in logic. In C. Casadio and P.J. Scott, editors, *Joachim Lambek: The Interplay of Mathematics, Logic, and Linguistics.* Springer, 2021. arXiv:2001.09195.

[Bor94] F. Borceux. *Handbook of Categorical Algebra II. Categories and Structures*, volume 51 of *Encyclopedia of Mathematics and Its Applications.* Cambridge University Press, 1994.

[But98] C. Butz. Regular categories and regular logic. BRICS Lecture Series, 1998.

[Dyb96] P. Dybjer. Internal type theory. *LNCS*, 1158:120–134, 1996.

[Fre72] Peter Freyd. Aspects of topoi. *Bulletin of the Australian Mathematical Society*, 7:1–76, 1072.

[GER96] Houman Zolfaghari Gonzalo E. Reyes. Bi-heyting algebras, toposes and modalities. *J. Phi. Logic*, 25:25–43, 1996.

[Hof] Martin Hofmann. Syntax and semantics of dependent types. In *Semantics and logics of computation.*

[HS98] Martin Hofmann and Thomas Streicher. The groupoid interpretation of type theory. In *Twenty-five years of constructive type theory (Venice, 1995)*, volume 36 of *Oxford Logic Guides*, pages 83–111. Oxford Univ. Press, New York, 1998.

[Joh03] P.T. Johnstone. *Sketches of an Elephant: A Topos Theory Compendium, 2 vol.s.* Number 43 in Oxford Logic Guides. Oxford University Press, 2003.

[Law69] F.W. Lawvere. Adjointness in foundations. *Dialectica*, 23:281–296, 1969.

[Law70] F.W. Lawvere. Equality in hyperdoctrines and comprehension schema as an adjoint functor. *Proceedings of the AMS Symposium on Pure Mathematics XVII*, pages 1–14, 1970.

[Law91] F. W. Lawvere. Intrinsic co-heyting boundaries and the leibniz rule in certain toposes. In G. Rosolini A. Carboni, M. Pedicchio, editor, *Category Theory - Como 1990*, number 1488 in LNM. Springer-Verlag, Heidelberg, 1991.

[Mak87] Michael Makkai. Stone duality for first order logic. *Adv. Math.*, 65:97–170, 1987.

[Mak93] Michael Makkai. *Duality and Definability*, volume 503. AMS, 1993.

[MH92] Michael Makkai and Victor Harnik. Lambek's categorical proof theory and Läuchli's abstract realizability. *Journal of Symbolic Logic*, 57(1):200–230, 1992.

[ML84] Per Martin-Löf. *Intuitionistic type theory*, volume 1 of *Studies in Proof Theory.* Bibliopolis, 1984.

[MM92] S. Mac Lane and I. Moerdijk. *Sheaves in Geometry and Logic. A First Introduction to Topos Theory.* Springer-Verlag, New York, 1992.

[MR95] Michael Makkai and Gonzalo Reyes. Completeness results for intuitionistic and modal logic in a categorical setting. *Annals of Pure and Applied Logic*, 72:25–101, 1995.

[Pal03] Erik Palmgren. Groupoids and local cartesian closure. 08 2003. unpublished.

[PV07] E. Palmgren and S.J. Vickers. Partial horn logic and cartesian categories. *Annals of Pure and Applied Logic*, 145(3):314–353, 2007.

[Sco70] Dana S. Scott. Constructive validity. In M. Laudet, D. Lacombe, L. Nolin, and M. Schützenberger, editors, *Symposium on Automatic Demonstration*, volume 125, pages 237–275. Springer-Verlag, 1970.

[Sco80] Dana S. Scott. Relating theories of the lambda calculus. In J. Roger Seldin, Jonathan P.; Hindley, editor, *To H.B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*, pages 403–450. Academic Press, 1980.

[See84] R. A. G. Seely. Locally cartesian closed categories and type theory. *Mathematical Proceedings of the Cambridge Philosophical Society*, 95(1):33, 1984.

[Tai68] William W. Tait. Constructive reasoning. In *Logic, Methodology and Philos. Sci. III (Proc. Third Internat. Congr., Amsterdam, 1967)*, pages 185–199. North-Holland, Amsterdam, 1968.