Lecture Notes:
Introduction to Categorical Logic

[DRAFT: January 15, 2003]

Steven Awodey Andrej Bauer

January 15, 2003

Contents

Foreword

These lecture notes were written in the Fall of 2002 for two introductory courses in categorical logic—the first author gave such a course at the Department of Philosophy at Carnegie Mellon University, and at the same time the second author gave a very similar course at the Department of Mathematics and Physics at University of Ljubljana. This was the third time such a course was given at Carnegie Mellon University, and so the material was deemed mature enough to be written up as lecture notes.

The course is intended for advanced undergraduate students and graduate students who have some background in category theory. We expect that students who have heard and used basic concepts of category theory in undergraduate courses, such as algebraic topology and abstract algebra, will be able to follow the material. We hope that the extensive review of category theory in Chapter **??** will also allow motivated students who lack category-theoretic background to quickly catch up.

Categorical logic is a broad subject, and so an introductory course must necessarily make certain choices. From a desire to keep the required category-theoretic machinery at a minimum we avoided any use of fibered categories. Throughout, our motto was "types are objects, propositions are subobjects". From a technical point of view this approach leads to problems when one considers dependent types, but from an educational point of view it is far outweighed by the simpler setup and transparency of ideas. In particular, we present the semantics of dependent types in terms of locally cartesian categories, and only comment briefly on problems involving equality of types that arise in this kind of semantics.

If you find any mistakes in the notes, and undoubtedly there are some, please contact us. An updated version of the notes and a list of known mistakes is available at `http://andrej.com/catlog/`.

Steve Awodey and Andrej Bauer

Chapter 2

Type Theories

2.1 Algebraic Theories

In this section we study a general approach to algebraic structures such as groups, rings, modules, and lattices. These are characterized by axiomatizations which involve only constants, operations, and equations. It is important that the operations are defined everywhere, which excludes two important examples: fields because the inverse of 0 is undefined, and categories because composition is defined only for some pairs of morphisms.

Let us start with the quintessential algebraic theory—the theory of a group. A group is a set G with a binary operation $\cdot : G \times G \to G$, satisfying the two axioms

$$\forall\, x, y, z{:}G\,.\,(x \cdot y) \cdot z = x \cdot (y \cdot z)$$
$$\exists\, e{:}G\,.\,\forall\, x{:}G\,.\,\exists\, y{:}G\,.\,(e \cdot x = x \cdot e = x \wedge x \cdot y = y \cdot x = e)$$

We want to pay attention to the logical form of the axioms, because axioms with a simpler logical structure can be interpreted more widely. The second axiom, which expresses the existence of a unit and inverse elements, is particularly unsatisfactory because it involves nested quantifiers.

If we require the unit to be a distinguished *constant* $e \in G$ and the inverse to be an *operation* ${}^{-1} : G \to G$ we obtain an equivalent formulation in which all axioms are *equations*: It is understood that equality is an equivalence relation and that we may substitute equals for equals. Notice that the universal quantifier is not needed anymore, provided we interpret the variables as ranging freely over G. In fact, we do not need to explicitly mention the underlying set G at all. Additionally, a constant can be thought of as a nullary operation, i.e., a function $1 \to G$. This leads to the general definition of an algebraic theory.

Definition 2.1.1 A *signature* Σ for an algebraic theory consists of a family of sets $\{\Sigma_k\}_{k \in \mathbb{N}}$. The elements of Σ_k are called the *k-ary operations*. In particular, the elements of Σ_0 are the *nullary operations* or *constants*.

The *terms* of a signature Σ are expressions constructed inductively by the following rules:

1. variables x, y, z, ... , are terms,
2. if $\langle t_1, \ldots, t_k \rangle$ is a k-tuple of terms and $f \in \Sigma_k$ is a k-ary operation then $f\langle t_1, \ldots, t_k \rangle$ is a term.

Definition 2.1.2 (cf. Definition 2.1.13) An *algebraic theory* $\mathbb{A} = (\Sigma, A)$ is given by a signature Σ and a set A of *axioms*, which are equations between terms.

Algebraic theories are also called *equational theories* and *Lawvere theories*.

Example 2.1.3 The theory of a commutative ring with unit is an algebraic theory. There are two nullary operations (constants) 0 and 1, a unary operation $-$, and two binary operations $+$ and $\cdot$. The equations are:

$$\begin{aligned} (x+y)+z &= x+(y+z) & (x\cdot y)\cdot z &= x\cdot(y\cdot z)\\ x+0 &= x & x\cdot 1 &= x\\ 0+x &= x & 1\cdot x &= x\\ x+(-x) &= 0 & (x+y)\cdot z &= x\cdot z + y\cdot z\\ (-x)+x &= 0 & z\cdot(x+y) &= z\cdot x + z\cdot y\\ x+y &= y+x & x\cdot y &= y\cdot x \end{aligned}$$

Example 2.1.4 The theory with no operations and no equations is the theory of a set.

Example 2.1.5 The theory with one constant and no equations is the theory of a *pointed set*, cf. Example **??**.

Example 2.1.6 Let R be a ring. A left R-module is an algebraic theory. It has one constant 0, a unary operation $-$, a binary operation $+$, and for each $a \in R$ a unary operation $\overline{a}$, called *scalar multiplication by* a. The following equations hold:

$$\begin{aligned} (x+y)+z &= x+(y+z)\,, & x+y &= y+x\,,\\ x+0 &= x\,, & 0+x &= x\,,\\ x+(-x) &= 0\,, & (-x)+x &= 0\,. \end{aligned}$$

For every $a, b \in R$ we also have the equations

$$\overline{a}(x+y) = \overline{a}\,x + \overline{a}\,y\,, \qquad \overline{a}(\overline{b}\,x) = \overline{(ab)}\,x\,, \qquad \overline{(a+b)}\,x = \overline{a}\,x + \overline{b}\,x\,.$$

Scalar multiplication by a is usually written as $a \cdot x$ instead of $\overline{a}\,x$. If we replace the ring R by a field $\mathbb{F}$ we obtain an algebraic theory of a vector space over $\mathbb{F}$, even though the theory of fields is not algebraic.

2.1.1 Many-sorted algebraic theories

It is sometimes necessary to consider algebraic theories with more than one set. In Example 2.1.6 we saw that the theory of a left R-module is algebraic, for a fixed ring R. However, if we wanted to have a general theory of left modules, we would need an algebraic structure consisting of *two* carrier sets, a ring R and a module M, together with operations and equations. To cover such examples we would have to consider *many-sorted algebraic theories*. Here we only give several motivating examples of many-sorted algebraic theories, and postpone the general study of many-sorted theories to subsequent sections.

Example 2.1.7 As already mentioned, the theory of left modules is a two-sorted algebraic theory. There are two sorts, R and M. The axioms express the fact that R is a ring, M is a commutative group, and that scalar multiplication has the desired properties, cf. Example 2.1.6.

Example 2.1.8 The theory of a directed graph is a two-sorted algebraic theory with a sort E for edges and a sort V for vertices. There are two unary operations $\mathtt{src} : E \to V$ and $\mathtt{trg} : E \to V$, which give the source and the target of an edge. There are no equations.

A *symmetric graph* is a graph in which for every edge $e : a \to b$ there is an edge $\overline{e} : b \to a$ in the other direction. This can be axiomatized with a unary operation $e \mapsto \overline{e}$ which satisfies the axioms

$$\mathsf{src}\,\overline{e} = \mathsf{trg}\,e\,, \qquad\qquad \mathsf{trg}\,\overline{e} = \mathsf{src}\,e\,.$$

A *simple graph* is one in which every two vertices are connected with at most one edge. An axiom which expresses this fact is

$$(\mathsf{src}\,e = \mathsf{src}\,f \wedge \mathsf{trg}\,e = \mathsf{trg}\,f) \Longrightarrow e = f\,.$$

However, this axiom has the form of an implication, so it seems that the theory of a simple graph is not algebraic.

Example 2.1.9 Modern computers are equipped with *random access memory (RAM)*. An idealized RAM consists of a number of *memory locations* indexed by a set A of *addresses*. Each memory location contains *data* which is an element of a set D. For example, the addresses might be 64-bit integers and data might be 8-bit integers.[1] The two basic operations on memory are a *memory lookup* and a *memory update*.

Let M be the set of all possible memory configurations. For each address $a \in A$ we have an operation $\mathtt{lookup}_a : M \to D$ which returns the content of location a, and an update operation $\mathtt{update}_a : M \times D \to M$ which updates the content of location a and returns the updated memory configuration. These

[1] An n-bit integer is an integer between 0 and $2^n - 1$.

operations satisfy the following equations, for all $m \in M$, $d, e \in D$, and $a, b \in A$ with $a \neq b$:

$$\begin{aligned}
\mathtt{lookup}_a\,(\mathtt{update}_a\,\langle m, d\rangle) &= d \\
\mathtt{lookup}_a\,(\mathtt{update}_b\,\langle m, d\rangle) &= \mathtt{lookup}_a\,m \\
\mathtt{update}_a\,\langle \mathtt{update}_b\,\langle m, e\rangle, d\rangle &= \mathtt{update}_b\,\langle \mathtt{update}_a\,\langle m, d\rangle, e\rangle \\
\mathtt{update}_a\,\langle \mathtt{update}_a\,\langle m, d\rangle, e\rangle &= \mathtt{update}_a\,\langle m, e\rangle
\end{aligned}$$

We see that random access memory can be formalized as a *two-sorted algebraic theory* with a sort M of memory configurations and a sort D of data. The addresses A are *not* a sort, they just parameterize the lookup and update operations. Had we made A into a sort, we would encounter problems because the second and the third axioms above would have to express the condition $a \neq b$, which cannot be done with equations in general.

2.1.2 Models of Algebraic Theories

Let us now consider what a *model* of an algebraic theory is. In classical algebra, a group is given by a set G, an element $e \in G$, a function $m : G \times G \to G$ and a function $i : G \to G$, satisfying the group axioms:

$$\begin{aligned}
m\langle x, m\langle y, z\rangle\rangle &= m\langle m\langle x, y\rangle, z\rangle\ , \\
m\langle x, i\,x\rangle &= m\langle i\,x, x\rangle = e\ , \\
m\langle x, e\rangle &= m\langle e, x\rangle = x\ .
\end{aligned}$$

We would like to generalize this notion so that we can speak of models of group theory in categories other than Set. This can be accomplished by translating everything into the language of category theory: a group is given by an object $G \in \mathsf{Set}$ and three morphisms

$$e : 1 \to G\ , \qquad m : G \times G \to G\ , \qquad i : G \to G\ .$$

Associativity of m is expressed by commutativity of the following diagram:

$$\begin{array}{ccc}
G \times G \times G & \xrightarrow{m \times \pi_2} & G \times G \\
{\scriptstyle \pi_0 \times m}\downarrow & & \downarrow{\scriptstyle m} \\
G \times G & \xrightarrow[m]{} & G
\end{array}$$

Similarly, the axioms for the unit and the inverse are expressed by commutativity of the following diagrams:

$$\begin{array}{ccccc}
G \times 1 & \xrightarrow{1_G \times e} & G \times G & \xleftarrow{e \times 1_G} & 1 \times G \\
& {\scriptstyle \pi_0}\searrow & \downarrow{\scriptstyle m} & \swarrow{\scriptstyle \pi_1} & \\
& & G & &
\end{array}
\qquad\qquad
\begin{array}{ccccc}
G & \xrightarrow{\langle 1_G, i\rangle} & G \times G & \xleftarrow{\langle i, 1_G\rangle} & G \\
{\scriptstyle !_G}\downarrow & & \downarrow{\scriptstyle m} & & \downarrow{\scriptstyle !_G} \\
1 & \xrightarrow[e]{} & G & \xleftarrow[e]{} & 1
\end{array}$$

We see that this formulation makes sense in any category $\mathcal{C}$ with finite products.

In general, an *interpretation* I of the terms of a theory $\mathbb{A}$ in a category $\mathcal{C}$ is given by an object $I\mathbb{A} \in \mathcal{C}$ and, for each basic operation f of arity k, a morphism $If : (I\mathbb{A})^k \to I\mathbb{A}$. In particular, basic constants are interpreted as morphisms $1 \to I\mathbb{A}$. A general term t is always interpreted together with a *context* of variables $x_1, \ldots, x_n$, where the variables appearing in t must be among the variables appearing in the context. We write

$$x_1, \ldots, x_n \mid t \tag{2.1}$$

to indicate that the term t is to be understood in context $x_1, \ldots x_n$. The interpretation of (2.1) is a morphism $It : (I\mathbb{A})^n \to I\mathbb{A}$, determined by the following rules:

1. The interpretation of a variable x_i is the i-th projection $\pi_i : (I\mathbb{A})^n \to I\mathbb{A}$.
2. A term of the form $f\langle t_1, \ldots, t_k\rangle$ is interpreted as the composition

$$(I\mathbb{A})^n \xrightarrow{\langle It_1, \ldots, It_k\rangle} (I\mathbb{A})^k \xrightarrow{\qquad If \qquad} \mathbb{A}$$

   where $It_i : (I\mathbb{A})^n \to I\mathbb{A}$ is the interpretation of the subterm t_i, for $i = 1, \ldots, k$, and If is the interpretation of the basic operation f.

It is clear from this that the interpretation of a term really depends on the context. For example, the term $f\, x_1$ is interpreted as a morphism $If : I\mathbb{A} \to I\mathbb{A}$ in context x_1, and as the morphism $If \circ \pi_1 : (I\mathbb{A})^2 \to I\mathbb{A}$ in the context x_1, x_2.

Suppose u and v are terms in context $x_1, \ldots, x_n$. Then we say that the equation $u = v$ is *satisfied* by the interpretation I if Iu and Iv are interpreted as the same morphism. In particular, suppose $u = v$ is an axiom of the theory, and let $x_1, \ldots, x_n$ be all the variables appearing in u and v. We say that I *satisfies the axiom* $u = v$ when $x_1, \ldots, x_n \mid u$ and $x_1, \ldots, x_n \mid u$ are interpreted as the same morphism by I.

Definition 2.1.10 (cf. Definition 2.1.18) A *model* M of an algebraic theory $\mathbb{A}$ in a category $\mathcal{C}$ with finite products is an interpretation of the theory that satisfies all the axioms of the theory.

Example 2.1.11 A model of the theory of a set in a category $\mathcal{C}$ with finite products is determined by an object $A \in \mathcal{C}$. In other words, the mathematicians who live in category $\mathcal{C}$ think of the objects of the category as ordinary sets.

Exercise 2.1.12 A model of group theory in Set is just an ordinary group. But we can also ask what a model of a group is in an arbitrary category with finite products. Determine what the models of a group are in the following categories: the category of finite sets FinSet, the category of topological spaces Top, the category of graphs Graph, and the category of groups Group.

Hint: Only the last case is tricky. Before thinking about it, prove the following lemma [?, Lemma 3.11.6]. Let G be a set provided with two binary operations $\cdot$ and $\star$ and a common unit e, so that $x \cdot e = e \cdot x = x \star e = e \star x = x$. Suppose the two operations commute, i.e., $(x \star y) \cdot (z \star w) = (x \cdot z) \star (y \cdot w)$. Then they coincide, are *commutative* and associative.

2.1.3 Algebraic theories as categories

The preceding account of models of algebraic theories is rather syntactic in nature. We would prefer an approach that emphasizes the algebraic point of view. The first step towards this is a *representation-free* notion of algebraic theories.

Let us consider group theory again. The usual axiomatization in terms of unit, multiplication and inverse is not the only possible one. For example, an alternative axiomatization in terms of the unit e and a binary operation $\odot$, called *double division*, can be given with a single axiom [?]:

$$(x \odot (((x \odot y) \odot z) \odot (y \odot e))) \odot (e \odot e) = z \ .$$

The usual group operations are related to right division as follows:

$$x \odot y = x^{-1} \cdot y^{-1} \ , \qquad x^{-1} = x \odot e \ , \qquad x \cdot y = (x \odot e) \odot (y \odot e) \ .$$

There may be various reasons why we prefer to work with one formulation of group theory rather than another, but this should not be reflected in the general idea of what is a group. We want to avoid particular choices of basic constants, operations, and axioms.[2] This is accomplished by a rather brute method—simply take *all* operations built from unit, multiplication, and inverse as basic, and *all* valid equations of group theory as axioms. We can even go one step further and collect all the operations into a category. We describe the construction of such a category for a general algebraic theory.

Let $\mathbb{A}$ be an algebraic theory. We construct a category, which is also denoted by $\mathbb{A}$, as follows. As objects we take sequences of variables, called *contexts*,

$$[x_1, \ldots, x_n] \ . \qquad (n \geq 0)$$

A morphism from $[x_1, \ldots, x_m]$ to $[x_1, \ldots, x_n]$ is an n-tuple $\langle t_1, \ldots, t_n \rangle$, where each t_k is a term of the theory whose variables are among $x_1, \ldots, x_m$. Two such morphisms $\langle t_1, \ldots, t_n \rangle$ and $\langle u_1, \ldots, u_n \rangle$ are equal if, and only if, the axioms of the theory imply that $t_k = u_k$ for every $k = 1, \ldots, n$.[3] The composition of

[2] This is akin to a basis-free theory of vector spaces: it is better to formulate the idea of a vector space without speaking explicitly of vector bases, even though every vector space has one. Without a doubt, vector bases are important, but they really are a *derived* concept.

[3] Strictly speaking, morphisms are *equivalence classes* of terms, where two terms are equivalent when the theory proves them to be equal. It is cumbersome to work with equivalence classes of terms, so we prefer to work directly with terms but keep in mind that equality between them is equality in the algebraic theory.

morphisms

$$\langle t_1, \ldots, t_m \rangle : [x_1, \ldots, x_k] \to [x_1, \ldots, x_m]$$
$$\langle u_1, \ldots, u_n \rangle : [x_1, \ldots, x_m] \to [x_1, \ldots, x_n]$$

is the morphism $\langle v_1, \ldots, v_n \rangle$ whose i-th component is obtained by simultaneously substituting in u_i the terms $t_1, \ldots, t_m$ for the variables $x_1, \ldots, x_m$:

$$v_i = u_i[t_1, \ldots, t_m / x_1, \ldots, x_m] \qquad (1 \leq i \leq n)$$

The identity morphism on $[x_1, \ldots, x_n]$ is $\langle x_1, \ldots, x_n \rangle$. Observe that the object $[x_1, \ldots, x_{n+m}]$ is the product of $[x_1, \ldots, x_n]$ and $[x_1, \ldots, x_m]$ so that all finite products exist. Furthermore, every object is a product of finitely many instances of the object $[x_1]$.

The category $\mathbb{A}$ contains precisely the same "algebraic" information as the theory $\mathbb{A}$ which it was built from. Thus we are lead to the following alternative definition.

Definition 2.1.13 (cf. Definition 2.1.2) An *algebraic theory* $\mathbb{A}$ is a small category with finite products whose objects form a sequence $A^0, A^1, A^2, \ldots$ such that $A^m \times A^n = A^{m+n}$ for all $m, n \in \mathbb{N}$. In particular, $1 = A^0$ is the terminal object and every object is a product of finitely many copies of $A = A^1$.

An algebraic theory $\mathbb{A}$ in the sense of the above definition determines an algebraic theory in the sense of Definition 2.1.2 as follows. As basic operations with arity k we take the morphisms $A^k \to A$. There is a canonical interpretation in $\mathbb{A}$ of terms built from variables and morphisms $A^k \to A$, namely each morphism is interpreted by itself. An equation $u = v$ is taken as an axiom of the theory $\mathbb{A}$ if the canonical interpretations of u and v coincide.

The new view of algebraic theories immediately suggest some interesting examples.

Example 2.1.14 The algebraic theory $\mathcal{C}^\infty$ of smooth maps is the category whose objects are n-dimensional Euclidean spaces 1, $\mathbb{R}$, $\mathbb{R}^2$, ..., and whose morphisms are $\mathcal{C}^\infty$-maps between them. Recall that a $\mathcal{C}^\infty$-map $f : \mathbb{R}^n \to \mathbb{R}$ is a function which has all higher partial derivatives, and that a function $f : \mathbb{R}^n \to \mathbb{R}^m$ is a $\mathcal{C}^\infty$-map when its compositions $\pi_k \circ f : \mathbb{R}^n \to \mathbb{R}$ with projections $\pi_k : \mathbb{R}^m \to \mathbb{R}$ are $\mathcal{C}^\infty$-maps.

Example 2.1.15 Recall that a *(total) recursive function* $f : \mathbb{N}^m \to \mathbb{N}^n$ is one that can be computed by a Turing machine. This means that there exists a Turing machine which on input $\langle a_1, \ldots, a_m \rangle$ outputs the value of $f\langle a_1, \ldots, a_m \rangle$. The algebraic theory Rec of recursive functions is the category whose objects are finite powers of the natural numbers 1, $\mathbb{N}$, $\mathbb{N}^2$, ..., and whose morphisms are recursive functions between them. The reason for considering this theory is that its models in a category $\mathcal{C}$ with finite products give a a theory of computability in $\mathcal{C}$, cf. Example 2.1.24.

Example 2.1.16 In a category $\mathcal{C}$ with finite products every object $A \in \mathcal{C}$ determines a full subcategory consisting of the finite powers $1, A, A^2, A^3, \ldots$ and morphisms between them. This is the theory of the object A.

Exercise 2.1.17 In Example 2.1.4 we saw that the theory $\mathbb{S}$ with no operations and no axioms is the theory of a set. Prove that the corresponding category $\mathbb{S}$ is equivalent to $\mathsf{FinSet}^{\mathsf{op}}$ where FinSet is the category of finite sets and functions.

2.1.4 Models of Algebraic Theories as Functors

Having successfully turned the notion of an algebraic theory into a special kind of category, we naturally want to know what we can do about the notion of a model.

Suppose $\mathbb{A}$ is a theory and M is a model of $\mathbb{A}$ in a category $\mathcal{C}$. Then the interpretation M determines a functor $M : \mathbb{A} \to \mathcal{C}$ from the corresponding category $\mathbb{A}$, defined on objects by

$$M[x_1, \ldots, x_k] = (M\mathbb{A})^k \ ,$$

and on morphisms by the following rules:

1. The morphism $\langle x_i \rangle : [x_1, \ldots, x_k] \to [x_1]$ is mapped to the i-th projection $\pi_i : (M\mathbb{A})^k \to M\mathbb{A}$.

2. The morphism

   $$\langle f\langle t_1, \ldots, t_m\rangle\rangle : [x_1, \ldots, x_k] \to [x_1]$$

   is mapped to the composition

   $$(M\mathbb{A})^m \xrightarrow{\langle Mt_1, \ldots, Mt_m\rangle} (M\mathbb{A})^k \xrightarrow{\qquad Mf \qquad} \mathbb{A}$$

   where $Mt_i : (M\mathbb{A})^n \to M\mathbb{A}$ is the value of M on the morphisms $\langle t_i \rangle : [x_1, \ldots, x_k] \to [x_1]$, for $i = 1, \ldots, m$, and Mf is the interpretation of the basic operation f.

3. The morphism

   $$\langle t_1, \ldots, t_m\rangle : [x_1, \ldots, x_k] \to [x_1, \ldots, x_m]$$

   is mapped to the morphism $\langle Mt_1, \ldots, Mt_m\rangle$ where Mt_i is the value of M on the morphism $\langle t_i \rangle : [x_1, \ldots, t_k] \to [x_1]$.

That $M : \mathbb{A} \to \mathcal{C}$ really is a functor follows from the assumption that the interpretation M is a model, which means that all the equations of the theory are satisfied by it. Observe that the functor M is defined in such a way that it preserves finite products.

Suppose $N : \mathbb{A} \to \mathcal{C}$ is a functor from the category $\mathbb{A}$ that corresponds to the theory $\mathbb{A}$. When does it determine a model of the theory? Clearly, if N is to be a model of a theory, then it must interpret variables as projections, $N(x_i : [x_1, \ldots, x_n] \to [x_1]) = \pi_i$, which only makes sense if $N[x_1, \ldots, x_n] = (N[x_1])^n$. In other words, N must preserve finite products. But that is all that is required because functoriality of N guarantees that *all* valid equations of the theory are satisfied, so the axioms are certainly satisfied as well. Thus we see that models of algebraic theories are just finite products preserving functors.

Definition 2.1.18 (cf. Definition 2.1.10) A *model* of an algebraic theory $\mathbb{A}$ in a category $\mathcal{C}$ with finite products is a functor $M : \mathbb{A} \to \mathcal{C}$ which preserves finite products.

So far we have not spoken of *homomorphisms* between models of an algebraic theory. Now a suitable notion presents itself: since models are functors, homomorphisms between models are natural transformations.

Definition 2.1.19 Let $\mathbb{A}$ be an algebraic theory and let $\mathcal{C}$ be a category with finite products. The *category* $\mathsf{Mod}_{\mathcal{C}}(\mathbb{A})$ *of* $\mathbb{A}$*-models in* $\mathcal{C}$ has as objects functors $M : \mathbb{A} \to \mathcal{C}$ that preserve finite products and as morphisms natural transformations between such functors.

Definition 2.1.20 An *algebraic category* is a category that is equivalent to a category of models $\mathsf{Mod}_{\mathcal{C}}(\mathbb{A})$ of an algebraic theory.

Example 2.1.21 At the beginning of this section we mentioned that the theory of a field is not algebraic because inverse of 0 is undefined. In principle there could be an equivalent algebraic formulation of the theory of a field which would somehow circumvent this problem. We can now show that this is not the case by proving that the category Field of fields and field homomorphisms is not algebraic.

First observe that a category of models $\mathsf{Mod}_{\mathcal{C}}(\mathbb{A})$ has a terminal object because $\mathcal{C}$ has a terminal object 1 and the constant functor $\Delta_1 : \mathbb{A} \to \mathcal{C}$ which maps every context to 1 is a model. The functor Δ_1 is the terminal object in $\mathsf{Mod}_{\mathcal{C}}(\mathbb{A})$ because it is the terminal functor in the functor category $\mathcal{C}^{\mathbb{A}}$. Now in order to see that Field is not algebraic it is sufficient to show that there is no terminal field, which was Exercise **??**.

By the way, the solution to Exercise **??** goes as follows: if T were a terminal field then by considering the unique homomorphism $\mathbb{Z}_2 \to T$ we see that $1+1 = 0$ in T, and by the unique homomorphism $\mathbb{Z}_3 \to T$ we see that $1+1+1 = 0$ in T, from which we get the impossibility $1 = 0$.

Example 2.1.22 Let us see what the preceding definitions give us in the case of group theory $\mathbb{G}$. Recall that the category $\mathbb{G}$ consists of contexts $[x_1, \ldots, x_n]$

and terms built from variables and the basic group operations. A finite product preserving functor $M : \mathbb{G} \to \mathsf{Set}$ is then determined up to natural isomorphism by its action on the context $[x_1]$ and the terms representing the basic operations. If we set

$$G = M[x_1]\,, \qquad e = M(\cdot \mid e)\,,$$
$$i = M(x_1 \mid {x_1}^{-1})\,, \qquad m = M(x_1, x_2 \mid x_1 \cdot x_2)\,,$$

then (G, e, i, m) is a just a group with unit e, inverse i and multiplication m. That G satisfies the axioms for groups follows from functoriality of M. Conversely, any group (G, e, i, m) determines a finite product preserving functor $M_G : \mathbb{G} \to \mathsf{Set}$ defined by

$$M_G[x_1, \ldots, x_n] = G^n\,, \qquad M_G(\cdot \mid e)\,,$$
$$M_G(x_1 \mid {x_1}^{-1}) = i\,, \qquad M_G(x_1, x_2 \mid x_1 \cdot x_2) = m\,.$$

This suggests that $\mathsf{Mod}_{\mathsf{Set}}(\mathbb{G})$ is equivalent to Group, provided both categories have the same notion of morphisms.

Suppose then that (G, e_G, i_G, m_G) and (H, e_H, i_H, m_H) are groups, and let $\phi : M_G \Longrightarrow M_H$ be a natural transformation between the corresponding functors. Then ϕ is already determined by its component at $[x_1]$ because by naturality the following diagram commutes, for $1 \leq k \leq n$:

$$\begin{array}{ccc} G^n & \xrightarrow{\phi_{[x_1,\ldots,x_n]}} & H^n \\ {\scriptstyle G\pi_k = \pi_k}\downarrow & & \downarrow{\scriptstyle H\pi_k = \pi_k} \\ G & \xrightarrow[\phi_{[x_1]}]{} & H \end{array}$$

If we write $\phi' = \phi_{[x_1]}$ then it follows that $\phi_{[x_1,\ldots,x_n]} = \phi' \times \cdots \times \phi'$. Again, by naturality of ϕ we see that the following diagram commutes:

$$\begin{array}{ccc} G \times G & \xrightarrow{\phi' \times \phi'} & H \times H \\ {\scriptstyle m_G}\downarrow & & \downarrow{\scriptstyle m_h} \\ G & \xrightarrow[\phi']{} & H \end{array}$$

Similar commutative squares show that ϕ' preserves the unit and commutes with the inverse operation, therefore $\phi' : G \to H$ is indeed a group homomorphism. Conversely, a group homomorphism $\psi' : G \to H$ determines a natural transformation $\psi : G \Longrightarrow H$ whose component at $[x_1, \ldots, x_n]$ is the n-fold product $\psi' \times \cdots \psi' : G^n \to H^n$. This demonstrates that

$$\mathsf{Mod}_{\mathsf{Set}}(\mathbb{G}) \simeq \mathsf{Group}\,.$$

Example 2.1.23 Consider the theory $\mathcal{C}^\infty$ of smooth maps from Example 2.1.14. A model of this theory in Set is a finite product preserving functor $A : \mathcal{C}^\infty \to \mathsf{Set}$. Up to natural isomorphism it can be described as follows. A $\mathcal{C}^\infty$-model is given by a set A and for every smooth map $f : \mathbb{R}^n \to \mathbb{R}$ a function $Af : A^n \to A$ such that if $f : \mathbb{R}^n \to \mathbb{R}$ and $g_i : \mathbb{R}^m \to \mathbb{R}$, $i = 1, \ldots, n$, are smooth maps then, for all $a_1, \ldots, a_m \in A$,

$$Af((Ag_1)\langle a_1, \ldots, a_m\rangle, \ldots, (Ag_n)\langle a_1, \ldots, a_m\rangle) = A(f \circ \langle g_1, \ldots, g_n\rangle)\langle a_1, \ldots, a_m\rangle \; .$$

In particular, since multiplication and addition are smooth maps, A is a commutative ring with unit. Such structures are known as $\mathcal{C}^\infty$-*rings*. Therefore, the models in Set of the theory of smooth maps are the $\mathcal{C}^\infty$-rings.

Example 2.1.24 In Example 2.1.15 we defined the algebraic theory Rec with objects the finite powers of $\mathbb{N}$ and morphisms recursive functions. We now consider the category of its set-theoretic models $\mathcal{R} = \mathsf{Mod}_{\mathsf{Set}}(\mathsf{Rec})$.

First, there is the "identity" model $I \in \mathcal{R}$, defined by $I\mathbb{N}^k = \mathbb{N}^k$ and $If = f$. Given any model $S \in \mathcal{R}$, its object part is determined by $S_1 = S\mathbb{N}$ since $S\mathbb{N}^k = S_1^k$. For every $n \in \mathbb{N}$ there is a morphism $1 \to \mathbb{N}$ in Rec defined by $\star \mapsto n$. Thus we have for each $n \in \mathbb{N}$ an element $s_n = S(\star \mapsto n) : 1 \to S_1$. This defines a function $s : \mathbb{N} \to S_1$ which in turn determines a natural transformation $\sigma : I \Longrightarrow S$ whose component at $\mathbb{N}^k$ is $s \times \cdots \times s : \mathbb{N}^k \to S_1^k$.

2.1.5 Completeness and Universal Models

In the last section of this chapter we summarize our method of studying algebraic theories. The same method will be used for studying other kinds of theories as well. We then conclude the chapter by proving that categorical semantics of algebraic theories is complete.

Categorical logic has two sides, the *logical* and the *categorical* one. The logical side is embodied in the notion of a *logical system*, which consists of four parts:

Type theory
: A logical system has a type theory, which is a calculus of types and terms. For algebraic theories the calculus of types is trivial, since there is only one type which is not even explicitly mentioned. The terms are built from variables and basic operations.

Logic
: A logical system has a logic. A variety of different kinds of logic can be considered. Algebraic theories have a very simple kind of logic that only involves equations and equational reasoning.

Theory
: A theory is given by basic types, basic terms, and axioms. The types and the terms must be expressed in the type theory of the system, and the axioms must be expressed in the logic of the system.

Interpretations and Models
: The type theory and logic of a logical system can be interpreted in a category with sufficiently rich structure. For algebraic theories we considered categories with finite products. The interpretation is *denotational*, which means that it is defined inductively on the structure of types, terms, and logical formulas. An interpretation of a theory is a *model* if it satisfies all the axioms of the theory.

The type theory or the logic of a logical system can be very simple. When the logic is restricted to just equations between terms we usually speak of a *type theory* rather than a logical system. When the type system is trivial, so that all terms have the same type, we speak of of a *single-sorted* logic. On the other end of the spectrum are complicated logical systems in which type theory and logic are intertwined, for example in first-order logic over a dependent type theory with subset and quotient types. It can also happen that the type-theoretic and logical parts are identified, for example in Martin-Löf type theory.

Complementary to the logical system is its *categorical semantics*:

Theories are categories
: From a theory we can construct a category which expresses essentially the same information as the theory but hides syntactic details. The structure of the category reflects the underlying type theory and logic. For example, algebraic theories are categories that are sequences of finite powers of an object.

Models are functors
: A model is a functor from a theory to a category with sufficiently rich structure. The requirement that all axioms of the theory must be satisfied by a model then translates to the requirement that the model is a functor and that it preserves the structure of the theory. For models of algebraic theories we only required that they preserve finite products, whereas functoriality ensured that all valid equations of the theory, hence also the axioms, were preserved.

Homomorphisms are natural transformations
: We obtain a notion of homomorphisms between models for free: since models are functors, homomorphisms are natural transformations between them. Homomorphisms between models of algebraic theories turned out to be the usual notion of morphisms that preserved the algebraic structure.

Completeness and universal models
: It is desirable for a categorical semantics to be *complete*, or to even have *universal models*. We explain what this means next.

Consider an algebraic theory $\mathbb{A}$ and an equation $u = v$ of the theory. If the equation can be proved from the axioms of the theory, then every model of the theory satisfies the equation. The converse statement is

"Every model of $\mathbb{A}$ satisfies $u = v$." $\Longrightarrow$ "$\mathbb{A}$ proves equation $u = v$." .

This property is called *semantic completeness*. Models of algebraic theories are semantically complete.

Theorem 2.1.25 (Completeness for algebraic theories) *Suppose $\mathbb{A}$ is an algebraic theory. Then there exists a category $\mathcal{A}$ and a model $U \in \mathsf{Mod}_{\mathcal{A}}(\mathbb{A})$, called the* universal model *for $\mathbb{A}$, with the property that, for every equation $u = v$ of the theory $\mathbb{A}$,*

$$\text{“}U \text{ satisfies } u = v\text{.”} \iff \text{“}\mathbb{A} \text{ proves } u = v\text{.”}$$

Therefore, categorical semantics of algebraic theories is complete.

Proof. The algebraic theory $\mathbb{A}$ is a category with finite products, so for the category $\mathcal{A}$ we can simply take $\mathbb{A}$ itself! The models of $\mathbb{A}$ in $\mathcal{A} = \mathbb{A}$ are functors $\mathbb{A} \to \mathbb{A}$ that preserve finite products. One such model is the identity functor $U = 1_{\mathbb{A}} : \mathbb{A} \to \mathbb{A}$. Clearly, the identity functor identifies two k-ary operations $f : A^k \to A$ and $g : A^k \to A$ if, and only if, $f = g$. ∎

Classically, when we speak of models of algebraic theories we have in mind models in Set. Therefore, it is a bit unsatisfactory that the universal model from the preceding theorem is not set-theoretic. Nevertheless, we can always find a universal model in a category of generalized sets, namely in a presheaf category.[4]

Proposition 2.1.26 *Let $\mathbb{A}$ be an algebraic theory. The Yoneda embedding $\mathsf{y} : \mathbb{A} \to \widehat{\mathbb{A}}$ is a universal model for $\mathbb{A}$.*

Proof. The Yoneda embedding $\mathsf{y} : \mathbb{A} \to \widehat{\mathbb{A}}$ preserves limits, and in particular finite products, hence it is a model of $\mathbb{A}$ in $\widehat{\mathbb{A}}$. Because it is a functor it satisfies all equations that are proved by $\mathbb{A}$, and because it is faithful it does not validate any equations that are not proved by $\mathbb{A}$. In other words, it is a universal model. ∎

Example 2.1.27 We consider group theory one last time. The universal group is a group that satisfies every equation that is satisfied by all groups, and no others. Let us describe it as a generalized set. Recall that the theory of a group is a category $\mathbb{G}$ whose objects are contexts $[x_1, \ldots, x_n]$, $n \in \mathbb{N}$. The carrier U of the universal group is the Yoneda embedding of the context with one variable,

$$U = \mathsf{y}[x_1] = \mathbb{G}(-, [x_1])\ .$$

This is a set parametrized by the objects of $\mathbb{G}$. For every $n \in \mathbb{N}$, we get a set $U_n = \mathbb{G}([x_1, \ldots, x_n], [x_1])$ that consists of all terms built from n variables, modulo equations of group theory—which is precisely the free group on n generators! Unit, inverse, and multiplication on U are defined at each stage U_n as the corresponding operations on the free group on n generators.

[4] Recall that the objects of a presheaf category $\mathsf{Set}^{\mathcal{C}^{\mathsf{op}}}$ are functors $\mathcal{C}^{\mathsf{op}} \to \mathsf{Set}$, which can be thought of as sets parametrized by objects of $\mathcal{C}$. In this sense they are generalized sets.

To summarize, the universal group is the free group on n-generators, where $n \in \mathbb{N}$ is a parameter.

Exercise 2.1.28 The universal group U is a functor $\mathbb{G}^{\mathsf{op}} \to \mathsf{Set}$. In the last example we described the object part of U. What is the action of U on morphisms?

Exercise 2.1.29 Let s be a term of group theory with variables $x_1, \ldots, x_n$. On one hand we can think of s as an element of the free group U_n, and on the other we can consider the interpretation of s in the universal group U, namely a natural transformation $Us : U^n \Longrightarrow U$. Suppose t is another term of group theory with variables $x_1, \ldots, x_n$. Show that $Us = Ut$ if, and only if, $s = t$ in the free group U_n.

2.2 Cartesian Closed Categories

2.2.1 Exponentials

We motivate the notion of exponentials with a couple of examples. Consider first the category Poset 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:

$$f \leq g \iff \forall\, x{:}P\,.\, fx \leq gx\,. \qquad (f, g : P \to Q)$$

We see that $\mathsf{Hom}(P, Q)$ is again an object of Poset, when equipped with a suitable order.

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

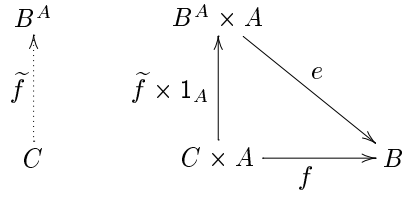
$$(f \cdot g)x = fx \cdot gx\,. \qquad (f, g : K \to M,\ x \in K)$$

On the other hand, in the category of groups there seems to be no natural way of defining a group structure on the set of all homomorphisms $\mathsf{Hom}(G, H)$ between groups G and H because not all homomorphisms are bijections.

These examples suggest that there ought to be a general notion of "exponentials" in a category. Speaking informally, the idea is that for objects A and B an exponential B^A is 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 $e : B^A \times A \to B$ which evaluates a morphism $f \in B^A$ at an argument $x \in A$ to give a value $e\langle f, x\rangle \in B$.

Definition 2.2.1 In a category $\mathcal{C}$ with binary products, an *exponential* (B^A, e) of objects A and B is an object B^A together with a morphism $e : B^A \times A \to B$, 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*[5] of f, for which the following diagram commutes:



The commutativity of the above diagram means that $f = e \circ (\widetilde{f} \times 1_A)$.

The above condition is called the *universal property of exponentials*. 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

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

That is all there is to it, really, except that variables and elements are never mentioned. The benefit of this is that the definition is completely general and applicable in categories whose objects are not sets.

In Poset the exponential Q^P of posets P and Q is the set of all monotone maps $P \to Q$, ordered pointwise. The evaluation map $e : Q^P \times P \to Q$ 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, $(\widetilde{f}z)x = f\langle z, x \rangle$. Therefore, Poset has all exponentials.

An object $A \in \mathcal{C}$ is *exponentiable* when the exponent B^A exists for every $B \in \mathcal{C}$. Sometimes, an object $B \in \mathcal{C}$ is called *baseable* when the exponent B^A exists for every $A \in \mathcal{C}$.

Proposition 2.2.2 *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

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

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

Proof. If such a right adjoint exists then the exponential of A and B is (B^A, ε_B), where $\varepsilon : -^A \times A \Longrightarrow 1_{\mathcal{C}}$ is the counit of the adjunction. The universal property of the exponential is precisely the universal property of the counit ε.

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

$$g^A = (g \circ \varepsilon_B)^{\sim} .$$

We use the formulation of adjoints by counit, cf. Proposition **??**, to show that $- \times A \dashv -^A$. The counit $\varepsilon : -^A \times A \Longrightarrow 1_{\mathcal{C}}$ at B is ε_B. The naturality square for ε,

$$\begin{array}{ccc} B^A \times A & \xrightarrow{\varepsilon_B} & B \\ {\scriptstyle f^A \times 1_A}\downarrow & & \downarrow{\scriptstyle f} \\ C^A \times A & \xrightarrow[\varepsilon_C]{} & C \end{array}$$

commutes because it is just the defining property of $(f \circ \varepsilon_B)^{\sim}$:

$$\varepsilon_C \circ (f^A \times 1_A) = \varepsilon_C \circ ((f \circ \varepsilon_B)^{\sim} \times 1_A) = f \circ \varepsilon_B .$$

The universal property of counit ε is precisely the universal property of the exponentials. ∎

Because exponentials are expressed as right adjoints to binary products, they are determined uniquely up to isomorphism.

Example 2.2.3 Consider propositional calculus P with conjunction and implication, as in Subsection **??**. Recall that P is the set of all propositions constructed from propositional variables, truth $\top$, falsehood $\bot$, conjunction $\wedge$, and implication $\Longrightarrow$. It is a preorder under the logical entailment relation $\vdash$. We saw already that implication is right adjoint to conjunction,

$$(- \times A) \dashv (A \Longrightarrow -) . \tag{2.2}$$

A conjunction $A \wedge B$ is a greatest lower bound of A and B, because we have, $A \wedge B \vdash A$, $A \wedge B \vdash B$, and for all propositions C,

$$\text{if } C \vdash A \text{ and } C \vdash B \text{ then } C \vdash A \wedge B.$$

Since in a preorder binary products are the same thing as greatest lower bounds, we see that conjunction is a binary product. Therefore, its right adjoint implication, is the exponential in P. The counit of the adjunction, or equivalently, the "evaluation" morphism, is the entailment

$$(A \Longrightarrow B) \wedge A \vdash B ,$$

which is the well known logical rule of *modus ponens*. This provides further evidence that concepts in logic arise as adjoints and their related notions.

Exercise 2.2.4 What is the unit of adjunction (2.2) in logical terms?

2.2.2 Cartesian Closed Categories

Definition 2.2.5 A *cartesian category* is a category that has finite products.

Definition 2.2.6 A *cartesian closed category (ccc)* is a category that has finite products and exponentials.

Equivalently, we could require the existence of the terminal object, binary products, and exponentials. The definition of cartesian closed categories can be phrased in terms of adjoint functors.

Proposition 2.2.7 *A category $\mathcal{C}$ is cartesian closed if, and only if, the following functors have* right *adjoints:*

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

Proof. Here $!_{\mathcal{C}}$ is the unique functor from $\mathcal{C}$ to the terminal category 1, and Δ is the diagonal functor,

$$\Delta A = \langle A, A \rangle \ , \qquad \Delta f = f \times f \ .$$

The right adjoint of $!_{\mathcal{C}}$ is a terminal object, the right adjoint of Δ is the binary product, and the right adjoint of $- \times A$ is exponentiation by A. ∎

We give a third formulation of cartesian closed categories, in terms of equations. 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 $\pi_0^{A,B} : A \times B \to A$ and $\pi_1^{A,B} : A \times B \to B$, 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 $e_{A,B} : B^A \times A \to B$, and a morphism $\widetilde{f} : C \to B^A$ for every morphism $f : C \times A \to B$.

We usually write π_0 and π_1 instead of $\pi_0^{A,B}$ and $\pi_1^{A,B}$. For morphisms $f : A \to B$ and $g : A' \to B'$ we define $f \times g : A \times A' \to B \times B'$ to be

$$f \times g = \langle f \circ \pi_0^{A,A'}, g \circ \pi_1^{A,A'} \rangle \ .$$

The types and morphisms satisfy the following equations:

1. For every $f : A \to 1$,

$$f = !_A \ .$$

2. For all $f : C \to A$, $g : C \to B$, $h : C \to A \times B$,

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

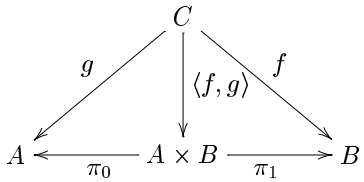
3. For all $f : C \times A \to B$, $g : C \to B^A$,

$$e_{A,B} \circ (\widetilde{f} \times 1_A) = f \ , \qquad (e_{A,B} \circ (g \times 1_A))^{\sim} = g \ .$$

These equations ensure that certain diagrams commute and that morphisms which 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:



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.

We now look at examples of cartesian closed categories.

Example 2.2.8 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 the set of all functions from X to Y,[6]

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

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

[6] In set theory, a function is the same thing as a functional relation.

Example 2.2.9 The category Cat of all small categories is cartesian closed. The exponential of small categories $\mathcal{C}$ and $\mathcal{D}$ is the functor category $\mathcal{D}^{\mathcal{C}}$, cf. isomorphism **??** on page **??**.

Example 2.2.10 A presheaf category $\widehat{\mathcal{C}}$ is cartesian closed, provided $\mathcal{C}$ is small. To see what the exponential of presheaves P and Q ought to be, we use Yoneda Lemma. If Q^P exists, then by Yoneda Lemma and the adjunction $(- \times P) \dashv (-^P)$, we have for all $A \in \mathcal{C}$,

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

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

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

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

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

Exercise 2.2.11 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 $\mathsf{Set}^{\cdot \rightrightarrows \cdot}$.

We have already seen that the exponential of posets P and Q is the poset Q^P of monotone functions from P to Q, ordered pointwise. Therefore Poset is cartesian closed. It is worth noting that even though the forgetful functor $U : \mathsf{Poset} \to \mathsf{Set}$ preserves finite limits, it does *not* preserve exponentials; in general $U(Q^P)$ is a proper subset of $(UQ)^{UP}$.

Exercise 2.2.12 There is a full and faithful functor $I : \mathsf{Set} \to \mathsf{Poset}$. Describe it and show that it preserves finite limits as well as exponentials.

Exercise 2.2.13 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 this adjoint is *not* the binary product in Vec.

Next we consider two important categories of cartesian closed posets—frames and Heyting algebras. They play an important role in logic and frames are important for topology, too.

2.2.3 Frames

A poset $(P, \leq)$, viewed as a category, is *cocomplete* when it has suprema (least upper bounds) of arbitrary subsets. This is so because coequalizers in a poset always exist, and coproducts are precisely least upper bounds. Recall that the supremum of $S \subseteq P$ is an element $\bigvee S \in P$ such that, for all $y \in S$,

$$\bigvee S \leq y \iff \forall\, x{:}S \,.\, x \leq y \;.$$

In particular, $\bigvee \emptyset$ is the least element of P and $\bigvee P$ is the greatest element of P. Similarly, a poset is *complete* when it has infima (greatest lower bounds) of arbitrary subsets; the infimum of $S \subseteq P$ is an element $\bigwedge S \in P$ such that, for all $y \in S$,

$$y \leq \bigwedge S \iff \forall\, x{:}S \,.\, y \leq x \;.$$

Proposition 2.2.14 *A poset is complete if, and only if, it is cocomplete.*

Proof. Infima and suprema are expressed in terms of each other as follows:

$$\bigwedge S = \bigvee \left\{ y \in P \;\middle|\; \forall\, x{:}S \,.\, y \leq x \right\} ,$$
$$\bigvee S = \bigwedge \left\{ y \in P \;\middle|\; \forall\, x{:}S \,.\, x \leq y \right\} .$$

■

Thus, we usually speak of *complete* posets only, even when we work with arbitrary suprema.

Suppose P is a complete poset. When is it cartesian closed? Being a complete poset, it has the terminal object, namely the greatest element $1 \in P$, and it has binary products which are binary infima. If P is cartesian closed then for all $x, y \in P$ there exists an exponential $(x \Rightarrow y) \in P$, which satisfies, for all $z \in P$,

$$\frac{z \wedge x \leq y}{z \leq x \Rightarrow y} \;.$$

With the help of this adjunction we derive the *infinite distributive law*, for an arbitrary family $\{y_i \in P \mid i \in I\}$,

$$x \wedge \bigvee_{i\in I} y_i = \bigvee_{i\in I}(x \wedge y_i) \tag{2.3}$$

as follows:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{x \wedge \bigvee_{i\in I} y_i \leq z}{\bigvee_{i\in I} y_i \leq (x \Rightarrow z)}}{\forall\, i{:}I\,.\,(y_i \leq (x \Rightarrow z))}}{\forall\, i{:}I\,.\,(x \wedge y_i \leq z)}}{\bigvee_{i\in I}(x \wedge y_i) \leq z}}$$

Now since $x \wedge \bigvee_{i\in I} y_i$ and $\bigvee_{i\in I}(x \wedge y_i)$ have the same upper bounds they must be equal.

Conversely, suppose the distributive law (2.3) holds. Then we can *define* $x \Rightarrow y$ to be

$$(x \Rightarrow y) = \bigvee \{z \in P \mid x \wedge z \leq y\} \ . \tag{2.4}$$

The best way to show that $x \Rightarrow y$ is the exponential of x and y is to use the characterization of adjoints by counit, as in Proposition **??**. In the case of $\wedge$ and $\Rightarrow$ this amounts to showing that, for all $x, y \in P$,

$$x \wedge (x \Rightarrow y) \leq y \ , \tag{2.5}$$

and that, for $z \in P$,

$$(x \wedge z \leq y) \Longrightarrow (z \leq x \Rightarrow y) \ .$$

This implication follows directly from (2.4), and (2.5) follows from the distributive law:

$$x \wedge (x \Rightarrow y) = x \wedge \bigvee \{z \in P \mid x \wedge z \leq y\} = \bigvee \{x \wedge z \mid x \wedge z \leq y\} \leq y \ .$$

Complete cartesian closed posets are called *frames*.

Definition 2.2.15 A *frame* is a poset that is complete and cartesian closed. Equivalently, a frame is a complete poset satisfying the distributive law

$$x \wedge \bigvee_{i\in I} y_i = \bigvee_{i\in I}(x \wedge y_i) \ .$$

A *frame morphism* is a function $f : L \to M$ between frames that preserves finite infima and arbitrary suprema. The category of frames and frame morphisms is denoted by Frame.

Example 2.2.16 The topology $\mathcal{O}X$ of a topological space X, ordered by inclusion, is a frame because finite intersections and arbitrary unions of open sets are open. The distributive law holds because intersections distribute over unions. If $f : X \to Y$ is a continuous map between topological spaces, the inverse image map $f^* : \mathcal{O}Y \to \mathcal{O}X$ is a frame homomorphism. Thus, there is a functor

$$\mathcal{O} : \mathsf{Top} \to \mathsf{Frame}^{\mathsf{op}}$$

which maps a space X to its topology $\mathcal{O}X$ and a continuous map $f : X \to Y$ to the inverse image map $f^* : \mathcal{O}Y \to \mathcal{O}X$.

The category $\mathsf{Frame}^{\mathsf{op}}$ is called the category of *locales* and is denoted by Loc. When we think of a frame as an object of Loc we call it a locale.

Exercise* 2.2.17 This exercise is meant for students with some background in topology. For a topological space X and a point $x \in X$, let $N(x)$ be the neighborhood filter of x,

$$N(x) = \left\{ U \in \mathcal{O}X \;\middle|\; x \in U \right\} .$$

Recall that a *T_0-space* is a topological space X in which points are determined by their neighborhood filters,

$$N(x) = N(y) \Longrightarrow x = y . \qquad (x, y \in X)$$

Let Top_0 be the full subcategory of Top on T_0-spaces. The functor $\mathcal{O} : \mathsf{Top} \to \mathsf{Loc}$ restricts to a functor $\mathcal{O} : \mathsf{Top}_0 \to \mathsf{Loc}$. Prove that $\mathcal{O} : \mathsf{Top}_0 \to \mathsf{Loc}$ is a faithful functor. Is it full?

2.2.4 Heyting Algebras

A *lattice* is a poset that has finite limits and colimits. In other words, a lattice $(L, \leq, \wedge, \vee, 0, 1)$ is a poset $(L, \leq)$ with distinguished elements $0, 1 \in L$, and binary operations meet $\wedge$ and join $\vee$, satisfying for all $x, y, z \in L$,

$$0 \leq x \leq 1 \qquad \frac{z \leq x \qquad z \leq y}{z \leq x \wedge y} \qquad \frac{x \leq z \qquad x \leq y}{x \vee y \leq z}$$

A *lattice homomorphism* is a function $f : L \to K$ between lattices which preserves finite limits and colimits, i.e., $f0 = 0$, $f1 = 1$, $f(x \wedge y) = fx \wedge fy$, and $f(x \vee y) = fx \vee fy$. The category of lattices and lattice homomorphisms is denoted by Lat.

A lattice can be axiomatized equationally as a set with two distinguished elements 0 and 1 and two binary operations $\wedge$ and $\vee$, satisfying the following

equations:

$$\begin{array}{ll} (x \wedge y) \wedge z = x \wedge (y \wedge z) \ , & (x \vee y) \vee z = x \vee (y \vee z) \ , \\ x \wedge y = y \wedge x \ , & x \vee y = y \vee x \ , \\ x \wedge x = x \ , & x \vee x = x \ , \\ 1 \wedge x = x \ , & 0 \vee x = x \ , \\ \multicolumn{2}{c}{x \wedge (y \vee x) = x = (x \wedge y) \vee x \ .} \end{array} \tag{2.6}$$

The partial order on L is then determined by

$$x \leq y \iff x \wedge y = x \ .$$

Exercise 2.2.18 Show that in a lattice $x \leq y$ if, and only if, $x \wedge y = x$ if, and only if, $x \vee y = y$.

A lattice is *distributive* if the following distributive laws hold in it:

$$\begin{array}{l} (x \vee y) \wedge z = (x \wedge z) \vee (y \wedge z) \ , \\ (x \wedge y) \vee z = (x \vee z) \wedge (y \vee z) \ . \end{array} \tag{2.7}$$

It turns out that if one distributive law holds then so does the other [?, I.1.5].

A *Heyting algebra* is a cartesian closed lattice H. This means that it has an operation $\Rightarrow$, satisfying for all $x, y, z \in H$

$$\frac{z \wedge x \leq y}{z \leq x \Rightarrow y}$$

A *Heyting algebra homomorphism* is a lattice homomorphism $f : K \to H$ between Heyting algebras that preserves implication, i.e., $f(x \Rightarrow y) = (fx \Rightarrow fy)$. The category of Heyting algebras and their homomorphisms is denoted by Heyt.

A Heyting algebra can be axiomatized equationally as a set H with two distinguished elements 0 and 1 and three binary operations $\wedge$, $\vee$ and $\Rightarrow$. The axioms for a Heyting algebra are the ones listed in (2.6), as well as the following ones for $\Rightarrow$:

$$\begin{array}{r} (x \Rightarrow x) = 1 \ , \\ x \wedge (x \Rightarrow y) = x \wedge y \ , \\ y \wedge (x \Rightarrow y) = y \ , \\ (x \Rightarrow (y \wedge z)) = (x \Rightarrow y) \wedge (x \Rightarrow z) \ . \end{array} \tag{2.8}$$

For a proof, see [?, I.1.10].

It turns out that every Heyting algebra is distributive [?, I.1.11].

Every frame is a Heyting algebra because it has all limits and colimits, therefore also the finite ones.

2.2.5 Intuitionistic Propositional Calculus

??

There is a forgetful functor $U : \mathsf{Heyt} \to \mathsf{Set}$ which maps a Heyting algebra to its underlying set, and it maps a morphism of Heyting algebras to the underlying function. Because Heyting algebras are an equational theory there is a left adjoint $H \dashv U$ which is the usual "free" construction that maps a set S to the free Heyting algebra HS generated by it. The construction of HS is performed in two steps: first we define a set HS of formal expressions, and then we quotient it by an equivalence relation generated by the axioms for Heyting algebras.

So let HS by the set of formal expressions generated inductively by the following rules:

1. Constants: $\bot, \top \in HS$.
2. Generators: if $x \in S$ then $x \in HS$.
3. Connectives: if $\phi, \psi \in HS$ then $\phi \wedge \psi, \phi \vee \psi, \phi \Rightarrow \psi \in HS$.

We impose an equivalence relation on HS, which we write as equality $=$ and think of it as such, to be the smallest equivalence relation satisfying axioms (2.6) and (2.8). This forces HS to be a Heyting algebra. We still need to define the action of H on morphisms: a function $f : S \to T$ is mapped to the Heyting algebra morphism $Hf : HS \to HT$ defined by

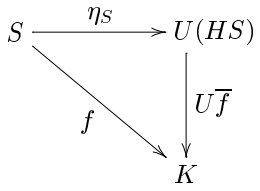
$$(Hf)\bot = \bot\,, \qquad (Hf)\bot = \bot\,, \qquad (Hf)x = fx\,,$$
$$(Hf)(\phi \star \psi) = ((Hf)\phi) \star ((Hf)\psi)\,,$$

where $\star$ stands for $\wedge$, $\vee$ or $\Rightarrow$.

There is an inclusion $\eta_S : S \to U(HS)$ of generators into the underlying set of HS. In fact we get a natural transformation $\eta : 1_{\mathsf{Set}} \Longrightarrow U \circ H$ which is the unit of the adjunction $H \dashv U$. To see this, consider a Heyting algebra K and an arbitrary function $f : S \to UK$. Then the Heyting algebra morphism $\overline{f} : HS \to K$ defined by

$$\overline{f}\bot = \bot\,, \qquad \overline{f}\bot = \bot\,, \qquad \overline{f}x = fx\,,$$
$$\overline{f}(\phi \star \psi) = (\overline{f}\phi) \star (\overline{f}\psi)\,,$$

where $\star$ stands for $\wedge$, $\vee$ or $\Rightarrow$, makes the following triangle commute:



It is a unique such morphism because any two morphisms from HS which agree on generators are equal. This is proved by induction on the structure of formal expressions in HS.

We may now define the *intuitionistic propositional calculus (IPC)* to be the free Heyting algebra IPC on countably many generators p_0, p_1, ... , called *atomic propositions* or *propositional variables*. This is a somewhat unorthodox definition from a logician's point of view—normally a *calculus* consists of a language, judgments, and rules of inference—but here we want to emphasize the idea that objects of interest, even when they are syntactically constructed, are characterized by their universal properties.

Having said that, let us also describe IPC in the usual way. The formulas of IPC are built inductively from propositional variables p_0, p_1, ... , constants falsehood $\bot$ and truth $\top$, and binary operations conjunction $\wedge$, disjunction $\vee$ and implication $\Rightarrow$. In IPC the basic judgment is *logical entailment*

$$u_1{:}A_1, \ldots, u_k{:}A_k \vdash B$$

which means "hypotheses A_1, ... , A_k entail proposition B". The hypotheses are labeled with distinct labels u_1, ... , u_k so that we can distinguish them, which is important when the same hypothesis appears more than once. Because the hypotheses are labeled it is irrelevant in what order they are listed, as long as the labels are not getting mixed up. Thus, the hypotheses $u_1{:}A \vee B, u_2{:}B$ are the same as the hypotheses $u_2{:}B, u_1{:}A \vee B$, but different from the hypotheses $u_1{:}B, u_2{:}A \vee B$. Sometimes we do not bother to label the hypotheses.

The left-hand side of a logical entailment is called the *context* and the right-hand side is the *conclusion*. Thus logical entailment is a relation between contexts and conclusions. The context may be empty. If Γ is a context, u is a label which does not occur in Γ, and A is a formula, then we write $\Gamma, u{:}A$ for the context Γ extended by the hypothesis $u{:}A$. Logical entailment is the smallest relation satisfying the following rules:

1. Conclusion from a hypothesis:

$$\frac{}{\Gamma \vdash A} \text{ if } u{:}A \text{ occurs in } \Gamma$$

2. Truth:

$$\frac{}{\Gamma \vdash \top}$$

3. Falsehood:

$$\frac{\Gamma \vdash \bot}{\Gamma \vdash A}$$

4. Conjunction:

$$\frac{\Gamma \vdash A \qquad \Gamma \vdash B}{\Gamma \vdash A \wedge B} \qquad \frac{\Gamma \vdash A \wedge B}{\Gamma \vdash A} \qquad \frac{\Gamma \vdash A \wedge B}{\Gamma \vdash B}$$

5. Disjunction:

$$\frac{\Gamma \vdash A}{\Gamma \vdash A \vee B} \qquad \frac{\Gamma \vdash B}{\Gamma \vdash A \vee B} \qquad \frac{\Gamma \vdash A \vee B \qquad \Gamma, u{:}A \vdash C \qquad \Gamma, v{:}B \vdash C}{\Gamma \vdash C}$$

6. Implication:

$$\frac{\Gamma, u{:}A \vdash B}{\Gamma \vdash A \Rightarrow B} \qquad \frac{\Gamma \vdash A \Rightarrow B \qquad \Gamma \vdash A}{\Gamma \vdash B}$$

A *proof* of $\Gamma \vdash A$ is a *finite* tree built from the above inference rules whose root is $\Gamma \vdash A$. For example, here is a proof of $A \vee B \vdash B \vee A$:

$$\dfrac{\dfrac{}{A \vee B \vdash A \vee B} \qquad \dfrac{\dfrac{}{A \vee B, A \vdash A}}{A \vee B, A \vdash B \vee A} \qquad \dfrac{\dfrac{}{A \vee B, B \vdash B}}{A \vee B, B \vdash B \vee A}}{A \vee B \vdash B \vee A}$$

We did not bother to label the hypotheses. A judgment $\Gamma \vdash A$ is *provable* if there exists a proof of it. Observe that every proof has at its leaves either the rule for $\top$ or a conclusion from a hypothesis.

You may wonder what happened to negation. In intuitionistic propositional calculus, negation is defined in terms of implication and falsehood as

$$\neg A \;\equiv\; A \Rightarrow \bot\,.$$

Properties of negation are then derived from the rules for implication and falsehood, see Exercise 2.2.22

Let P be the set of all formulas of IPC, preordered by the relation

$$A \vdash B\,, \qquad\qquad (A, B \in P)$$

where we did not bother to label the hypothesis A. Clearly, it is the case that $A \vdash A$. To see that $\vdash$ is transitive, suppose Π_1 is a proof of $A \vdash B$ and Π_2 is a proof of $B \vdash C$. Then we can obtain a proof of $A \vdash C$ from a proof Π_2 of $B \vdash C$ by replacing in it each use of the hypothesis B by the proof Π_1 of $A \vdash B$. This is worked out in detail in the next two exercises.

Exercise 2.2.19 Prove the following statement by induction on the structure of the proof Π: if Π is a proof of $\Gamma, u{:}A, v{:}A \vdash B$ then there is a proof of $\Gamma, u{:}A \vdash B$.

Exercise 2.2.20 Prove the following statement by induction on the structure of the proof Π_2: if Π_1 is a proof of $\Gamma \vdash A$ and Π_2 is a proof of $\Gamma, u{:}A \vdash B$, then there is a proof of $\Gamma \vdash B$.

Let IPC be the poset reflection of the preorder $(P, \vdash)$. The elements of IPC are equivalence classes $[A]$ of formulas, where two formulas A and B are equivalent if both $A \vdash B$ and $B \vdash A$ are provable. The poset IPC is just the free Heyting algebra on countably many generators p_0, p_1, ...

2.2.6 Boolean Algebras

An element $x \in L$ of a lattice L is said to be *complemented* when there exists $y \in L$ such that

$$x \vee y = 1 \,, \qquad\qquad x \wedge y = 0 \,.$$

We say that y is the *complement* of x.

In a distributive lattice, the complement of x is unique if it exists. Indeed, if both y and z are complements of x then

$$y \wedge z = (y \wedge z) \vee 0 = (y \wedge z) \vee (y \wedge x) = y \wedge (z \vee x) = y \wedge 1 = y \,,$$

hence $y \leq z$. A symmetric argument shows that $z \leq y$, therefore $y = z$. The complement of x, if it exists, is denoted by $\neg x$.

A *Boolean algebra* is a distributive lattice in which every element is complemented. In other words, a Boolean algebra B has the *complementation* operation $\neg$ which satisfies, for all $x \in B$,

$$x \wedge \neg x = 0 \,, \qquad\qquad x \vee \neg x = 1 \,. \tag{2.9}$$

The full subcategory of Lat consisting of Boolean algebras is denoted by Bool.

Exercise 2.2.21 Prove that every Boolean algebra is a Heyting algebra. Hint: how is implication encoded in terms of negation and disjunction in classical logic?

In a Heyting algebra not every element is complemented. However, we can still define a *pseudo complement* or *negation* operation $\neg$ by

$$\neg x = (x \Rightarrow 0) \,,$$

Then $\neg x$ is the largest element for which $x \wedge \neg x = 0$. While in a Boolean algebra $\neg\neg x = x$, in a Heyting algebra we only have $\neg\neg x \leq x$ in general. An element x of a Heyting algebra for which $\neg\neg x = x$ is called a *regular* element.

Exercise 2.2.22 Derive the following properties of negation in a *Heyting* algebra:

$$\begin{aligned} x &\leq \neg\neg x \,, \\ \neg x &= \neg\neg\neg x \,, \\ x \leq y &\Longrightarrow \neg y \leq \neg x \,, \\ \neg\neg(x \wedge y) &= \neg\neg x \wedge \neg\neg y \,. \end{aligned}$$

Exercise 2.2.23 The topology $\mathcal{O}X$ of a topological space X is a frame, therefore a Heyting algebra. Describe in topological language negation on $\mathcal{O}X$ and regular elements in $\mathcal{O}X$.

Exercise 2.2.24 Show that for a Heyting algebra H, the regular elements of H form a Boolean algebra $H_{\neg\neg} = \{x \in H \mid x = \neg\neg x\}$. Here $H_{\neg\neg}$ is viewed as a subposet of H. Hint: negation $\neg'$, conjunction $\wedge'$, and disjunction $\vee'$ in $H_{\neg\neg}$ are expressed as follows in terms of negation, conjunction and disjunction in H, for $x, y \in H_{\neg\neg}$:

$$\neg' x = \neg x \,, \qquad x \wedge' y = \neg\neg(x \wedge y) \,, \qquad x \vee' y = \neg\neg(x \vee y) \,.$$

The *classical propositional calculus (CPC)* is obtained from the intuitionistic propositional calculus by the addition of the logical rule known as *tertium non datur*, or the *law of excluded middle*:

$$\frac{}{\Gamma \vdash A \vee \neg A}$$

Alternatively, we could add the law known as *reductio ad absurdum*, or *proof by contradiction*:

$$\frac{\Gamma \vdash \neg\neg A}{\Gamma \vdash A} \,.$$

If we identify logically equivalent formulas of CPC we obtain a poset CPC ordered by logical entailment. This poset can be described by a universal property: it is the free Boolean algebra on countably many generators. The construction of a free Boolean algebra is performed just like the construction of a free Heyting algebra. The equational axioms for a Boolean algebra are the axioms for a lattice (2.6), the distributive laws (2.7), and the complement laws (2.9).

Exercise* 2.2.25 Is CPC isomorphic to the Boolean algebra $\mathsf{IPC}_{\neg\neg}$ of the regular elements of IPC?

Exercise 2.2.26 Show that in a Heyting algebra H, $\neg\neg x = x$ for all $x \in H$ if, and only if, $y \vee \neg y = 1$ for all $y \in H$. Hint: half of the equivalence is easy. For the other half, observe that the assumption $\forall\, x{:}H \,.\, \neg\neg x = x$ means that double negation is an order-reversing bijection $H \to H$. Therefore it transforms joins into meets and vice versa, and so *De Morgan laws* hold:

$$\neg(x \wedge y) = \neg x \vee \neg y \,, \qquad \neg(x \vee y) = \neg x \wedge \neg y \,.$$

De Morgan laws together with $y \wedge \neg y = 0$ easily imply $y \vee \neg y = 1$. See [**?**, I.1.11].

2.3 Simply Typed λ-calculus

The λ-calculus is the abstract theory of functions, just like group theory is the 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. 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

$$f x = t \,.$$

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 abstractions are nested:

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

This operation is called λ-*abstraction*.[7] For example, $\lambda x.\, \lambda y.\, (x + y)$ is the function which maps an argument a to the function $\lambda y.\, (a + y)$.

In an expression $\lambda x.\, t$ the variable x is *bound* in t. This means that we can rename it and we still have the same λ-abstraction. For example, $\lambda x.\, \lambda y.\, (x + y)$, $\lambda u.\, \lambda y.\, (u + y)$, and $\lambda u.\, \lambda x.\, (u + x)$ are all the same λ-abstraction. This is similar to bound variables in integrals, summations, and quantified formulas:

$$\int \frac{1}{1 + a x^2}\, dx \,, \qquad \sum_{k=1}^{\infty} \frac{1}{k^n} \,, \qquad \exists\, u{:}A \,.\, P(u, v) \,.$$

The bound variables are x, k, and u, respectively. The other variables, a, n and v, are *free*. If t is an expression then we denote by $\mathsf{FV}(t)$ the set of free variables of t. The fact that λ-abstraction binds a variable is expressed by the rules for computing $\mathsf{FV}(t)$:

$$\begin{aligned} \mathsf{FV}(x) &= \{x\} \qquad \text{if } x \text{ is a variable} \\ \mathsf{FV}(a) &= \emptyset \qquad \text{if } a \text{ is a constant} \\ \mathsf{FV}(t\, u) &= \mathsf{FV}(t) \cup \mathsf{FV}(u) \\ \mathsf{FV}(\lambda x.\, t) &= \mathsf{FV}(t) \setminus \{x\} \,. \end{aligned}$$

When we rename a bound variable, or substitute for a free one, we must be careful not to *capture* any variables. For example, if we want to substitute $1 + z$ for y in the expression

$$\lambda x.\, \lambda z.\, (x + y + z) \,,$$

then we must first rename the bound variable z, say to w, and only then substitute:

$$\lambda x.\, \lambda w.\, (x + 1 + z + w) \,.$$

[7] Why is the letter λ used? The notation goes back to Alonzo Church. We have it on good authority that once professor Church was sent a postcard asking him "Why λ?" He wrote the answer onto the same postcard and returned it. The answer was this: "enie menie miney mo".

If we substituted directly, we would get $\lambda x.\, \lambda z.\, (x + 1 + z + z)$, which is not what was intended because z was captured by the λ-abstraction.

There are two kinds of λ-calculus, the *typed* and the *untyped* one. 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 means. 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 f, a 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 $t \in A$. In these notes we will concentrate on the typed λ-calculus.

We now give a precise definition of what constitutes a *simply-typed λ-calculus*. First, we are given a set of *basic types*. We express the fact that A is a basic type with an axiom

$$\frac{}{A\ \texttt{type}}$$

which is read as "A is a type". There is a *unit type* $\texttt{1}$:

$$\frac{}{\texttt{1 type}}$$

We can also form *product types* and *function types*:

$$\frac{A\ \texttt{type} \qquad B\ \texttt{type}}{A \times B\ \texttt{type}} \qquad\qquad \frac{A\ \texttt{type} \qquad B\ \texttt{type}}{A \to B\ \texttt{type}}$$

To summarize, the set of all types is generated from the basic types and the unit type $\texttt{1}$ by formation of product and function types. Function types associate to the right:

$$A \to B \to C \equiv A \to (B \to C)\ .$$

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 the basic constants by the following grammar:

$$t ::= [\text{var}] \mid [\text{const}] \mid * \mid \langle t, t' \rangle \mid \texttt{fst}\, t \mid \texttt{snd}\, t \mid t\, t' \mid \lambda x{:}A\, .\, t$$

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 *free variables* $\mathsf{FV}(t)$ of a term t are computed as follows:

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

If u and t are terms and x is a variable, then we obtain a new term $t[u/x]$ by *substitution* of u for x in t. This means that we replace every *free* occurrence of x in t by u. If u has any free variables, we must make sure that they are not *captured* by the bound variables in t. This is accomplished by renaming the bound variables in t so that they are disjoint from the free variables in u. Assuming that this is the case, the rules for substitution are as follows:

$$\begin{aligned}
x[u/x] &= u\\
y[u/x] &= y \quad \text{if } x \neq y\\
a[u/x] &= a \quad \text{if } a \text{ is a basic constant}\\
\langle s, t\rangle[u/x] &= \langle s[u/x], t[u/x]\rangle\\
\mathtt{fst}\, t[u/x] &= \mathtt{fst}\, (t[u/x])\\
\mathtt{snd}\, t[u/x] &= \mathtt{snd}\, (t[u/x])\\
(s\, t)[u/x] &= (s[u/x])(t[u/x])\\
(\lambda y{:}A\,.\,t)[u/x] &= \lambda y{:}A\,.\,(t[u/x]) \quad \text{if } x \neq y \text{ and } y \notin \mathsf{FV}(u)
\end{aligned}$$

If $x_1, \ldots, x_n$ are *distinct* variables and $A_1, \ldots, A_n$ 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. Context 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 a has a uniquely determined type A,

$$\frac{}{\Gamma \mid a : A}$$

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:

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

The typing rules for application and λ-abstraction are:

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

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

$$\Gamma \mid u : A\,, \qquad\qquad \Gamma \mid t : B\,,$$

the judgment that they are equal is written as

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

Note that u 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:

1. Equality is an equivalence relation:

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

2. The weakening rule:

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

3. Unit type:

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

4. Equations for product types:

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

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

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

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

5. 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.

Apart from the above rules for equality we might want to impose additional equations. In this case we do not speak of a λ-calculus but rather of a λ-*theory*. Thus, a λ-theory $\mathbb{T}$ is given by a set of basic types, a set of basic constants, and a set of *equations* of the form

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

We summarize the preceding definitions.

Definition 2.3.1 A *simply-typed λ-calculus* is given by a set of *basic types* and a set of *basic constants* together with their types. *A simply-typed λ-theory* is a simply-typed λ-calculus together with a set of equations.

We use letters $\mathbb{S}$, $\mathbb{T}$, $\mathbb{U}$, ... to denote theories.

Example 2.3.2 The theory of a group is a simply-typed λ-theory. It has one basic type $\mathtt{G}$ and three basic constant, the unit $\mathtt{e}$, the inverse $\mathtt{i}$, and the group operation $\mathtt{m}$,

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

with the following equations:

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

These are just the familiar axioms for a group.

Example 2.3.3 In general, any algebraic theory $\mathbb{A}$ determines a λ-theory. 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 the terms of the corresponding λ-theory in a straightforward manner. For every axiom $t = u$ of $\mathbb{A}$ the corresponding axiom in the λ-theory is

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

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

Example 2.3.4 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 constant, source $\mathtt{src}$ and target $\mathtt{trg}$,

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

There are no equations.

Example 2.3.5 An example of a λ-theory is readily found in the theory of programming languages. The mini-programming language PCF is a simply-typed λ-calculus with a basic type $\mathtt{nat}$ for natural numbers, and a basic type $\mathtt{bool}$ of Boolean values,

$$\frac{}{\mathtt{nat\ type}} \qquad \frac{}{\mathtt{bool\ type}}$$

There are basic constants zero $\mathtt{0}$, successor $\mathtt{succ}$, the Boolean constants $\mathtt{true}$ and $\mathtt{false}$, comparison with zero $\mathtt{iszero}$, 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:

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

Example 2.3.6 Another example of a λ-theory is the *theory of a reflexive type*. This theory has one basic type $\mathtt{D}$ 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

$$f : \mathtt{D} \to \mathtt{D} \mid \mathtt{r}\,(\mathtt{s}\, f) = f : \mathtt{D} \to \mathtt{D} \tag{2.10}$$

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

$$x : \mathtt{D} \mid \mathtt{s}\,(\mathtt{r}\, x) = x : \mathtt{D} \tag{2.11}$$

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

2.3.1 Untyped λ-calculus

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

$$t ::= [\mathrm{var}] \mid t\, t' \mid \lambda x.\, t\ .$$

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:

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

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

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

The untyped λ-calculus can be translated into the theory of a reflexive type from Example 2.3.6. 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

$$\Gamma \mid u = t \tag{2.12}$$

is translated to the judgment

$$\Gamma^* \mid u^* = t^* : \mathtt{D}\ . \tag{2.13}$$

Exercise* **2.3.7** Prove that if equation (2.12) is provable then equation (2.13) is provable as well. Identify precisely at which point in your proof you need to use equations (2.10) and (2.11). Does provability of (2.13) imply provability of (2.12)?

2.4 Completeness of λ-calculus

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

1. For every basic type A in $\mathbb{T}$ an object $[\![A]\!] \in \mathcal{C}$. The interpretation is extended to all types by
$$[\![1]\!] = 1\ , \qquad [\![A \times B]\!] = [\![A]\!] \times [\![B]\!]\ , \qquad [\![A \to B]\!] = [\![B]\!]^{[\![A]\!]}\ .$$
2. For every basic constant c of type A a morphism $[\![c]\!] : 1 \to [\![A]\!]$.

The interpretation is extended to all terms in contexts 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 1. A typing judgment
$$\Gamma \mid t : A$$
is interpreted as a morphism
$$[\![\Gamma \mid t : A]\!] : [\![\Gamma]\!] \to [\![A]\!]\ .$$
The interpretation is defined inductively by the following rules:

1. The i-th variable is interpreted as the i-th projection,
$$[\![x_0{:}A_0, \ldots, x_n{:}A_n \mid x_i : A_i]\!] = \pi_i : [\![\Gamma]\!] \to [\![A_i]\!]\ .$$
2. A basic constant $c : A$ in context Γ is interpreted as the composition
$$[\![\Gamma]\!] \xrightarrow{!_{[\![\Gamma]\!]}} 1 \xrightarrow{[\![c]\!]} [\![A]\!]$$
3. The interpretation of projections and pairs is
$$\begin{aligned} [\![\Gamma \mid \langle t, u\rangle : A \times B]\!] &= \langle [\![\Gamma \mid t : A]\!], [\![\Gamma \mid u : B]\!]\rangle : [\![\Gamma]\!] \to [\![A]\!] \times [\![B]\!] \\ [\![\Gamma \mid \mathtt{fst}\, t : A]\!] &= \pi_0 \circ [\![\Gamma \mid t : A \times B]\!] : [\![\Gamma]\!] \to [\![A]\!] \\ [\![\Gamma \mid \mathtt{snd}\, t : A]\!] &= \pi_1 \circ [\![\Gamma \mid t : A \times B]\!] : [\![\Gamma]\!] \to [\![B]\!]\ . \end{aligned}$$

4. The interpretation of application and λ-abstraction is

$$[\![\Gamma \mid t\,u : B]\!] = e \circ \langle [\![\Gamma \mid t : A \to B]\!], [\![\Gamma \mid u : A]\!] \rangle : [\![\Gamma]\!] \to [\![B]\!]$$
$$[\![\Gamma \mid \lambda x{:}A\,.\,t : A \to B]\!] = ([\![\Gamma, x{:}A \mid t : B]\!])^{\sim} : [\![\Gamma]\!] \to [\![B]\!]^{[\![A]\!]}$$

where $e : [\![A \to B]\!] \times [\![A]\!] \to [\![B]\!]$ is the evaluation morphism for $[\![B]\!]^{[\![A]\!]}$ and $([\![\Gamma, x{:}A \mid t : B]\!])^{\sim}$ is the transpose of the morphism

$$[\![\Gamma, x{:}A \mid t : B]\!] : [\![\Gamma]\!] \times [\![A]\!] \to [\![B]\!] .$$

An interpretation of the λ-calculus of a theory $\mathbb{T}$ is a *model* of the theory if it satisfies all axioms of $\mathbb{T}$. This means that, for every axiom $\Gamma \mid t = u : A$, the interpretations of u and t coincide, $[\![\Gamma \mid u : A]\!] = [\![\Gamma \mid t : A]\!]$. It follows that all equations provable in $\mathbb{T}$ are satisfied in the model.

In Section 2.1 we saw that algebraic theories can be viewed as categories, cf. Definition 2.1.13, and models as functors, cf. Definition 2.1.18. The same can be done with λ-theories and their models. The first step is to build a category from a λ-theory.

Given a λ-theory $\mathbb{T}$, we construct a *syntactic category* $\mathcal{S}(\mathbb{T})$ as follows. The objects of $\mathcal{S}(\mathbb{T})$ are the types of $\mathbb{T}$. Morphisms $A \to B$ are terms in context

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

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

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

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

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

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

Proposition 2.4.1 *The syntactic category $\mathcal{S}(\mathbb{T})$ built from a λ-theory is cartesian closed.*

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

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

This morphism is unique because

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

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

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

Given morphisms

$$z : C \mid t : A \,, \qquad\qquad z : C \mid u : B \,,$$

the term

$$z : C \mid \langle t, u \rangle : A \times B$$

represents the unique morphism satisfying

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

Indeed, if $\mathtt{fst}\,s = t$ and $\mathtt{snd}\,s = u$ then

$$s = \langle \mathtt{fst}\,s, \mathtt{snd}\,s \rangle = \langle t, u \rangle \,.$$

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

$$p : (A \to B) \times A \;\big|\; (\mathtt{fst}\,p)(\mathtt{snd}\,p) : B \,.$$

The transpose of the morphism $p : C \times A \mid t : B$ is

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

Showing that this is the transpose of t amounts to

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

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

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

is equal to $\lambda x{:}A \,.\, (t[\langle z, x \rangle / p])$. First observe that

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

Therefore,

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

as required. ∎

The syntactic category allows us to define models as functors.

Definition 2.4.2 A *model* M of a λ-theory $\mathbb{T}$ in a cartesian closed category $\mathcal{C}$ is a functor $M : \mathcal{S}(\mathbb{T}) \to \mathcal{C}$ that preserves finite products and exponentials.

We can now proceed much as we did in the case of algebraic theories and prove that the semantics of λ-theories in cartesian closed categories is complete:

"$\mathbb{T}$ proves $\Gamma \mid t = u : A$." $\iff$ "Every model of $\mathbb{T}$ satisfies $\Gamma \mid t = u : A$."

This is proved exactly the same way as for algebraic theories. A λ-theory $\mathbb{T}$ has a *canonical interpretation* in the syntactic category $\mathcal{S}(\mathbb{T})$ which interprets a basic type A as itself, and a basic constant c of type A as the morphism $x : 1 \mid c : A$. The canonical interpretation is a model of $\mathbb{T}$, also known as the *syntactic model*. It is in fact a universal model for $\mathbb{T}$ because by construction of $\mathcal{S}(\mathbb{T})$, for any terms $\Gamma \mid u : A$ and $\Gamma \mid t : A$,

"$\mathbb{T}$ proves $\Gamma \mid u = t : A$" $\iff$ "Syntactic model satisfies $\Gamma \mid u = t : A$" .

We take one step further and organize λ-theories into a category. For this we need to define a suitable notion of morphisms. A *translation* $\tau : \mathbb{T} \to \mathbb{U}$ of a λ-theory $\mathbb{T}$ into a λ-theory $\mathbb{U}$ is given by the following data:

1. For each basic type A in $\mathbb{T}$ a type τA in $\mathbb{U}$. The translation is then extended to all types by the rules

$$\tau 1 = 1 \ , \qquad \tau(A \times B) = \tau A \times \tau B \ , \qquad \tau(A \to B) = \tau A \to \tau B \ .$$

2. For each basic constant c of type A in $\mathbb{A}$ a term τc of type τA in $\mathbb{U}$. 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 $\mathbb{T}$: if $\Gamma \mid t = u : A$ is an axiom of $\mathbb{T}$ then $\mathbb{U}$ proves $\tau\Gamma \mid \tau t = \tau u : \tau A$. It then follows that all equations proved by $\mathbb{T}$ are translated to valid equations in $\mathbb{U}$.

A moment of consideration shows that a translation $\tau : \mathbb{T} \to \mathbb{U}$ is the same thing as a model of $\mathbb{T}$ in $\mathcal{S}(\mathbb{U})$.

Clearly, λ-theories and translations between them 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. It corresponds to the canonical interpretation of $\mathbb{T}$ in $\mathcal{S}(\mathbb{T})$.

Definition 2.4.3 $\lambda\mathsf{Thr}$ is the category whose objects are λ-theories and morphisms are translations between them.

Let $\mathcal{C}$ be a small cartesian closed category. There is a λ-theory $\mathbb{L}(\mathcal{C})$ that corresponds to $\mathcal{C}$, called the *internal language of* $\mathcal{C}$, 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
$$x : A \mid t : B\,, \qquad\qquad y : B \mid u : A\,,$$
such that $\mathbb{T}$ proves
$$x : A \mid u[t/y] = x : A\,, \qquad\qquad y : B \mid t[u/x] = y : B\,.$$
Furthermore, an *equivalence of theories* $\mathbb{T}$ and $\mathbb{U}$ is a pair of translations
$$\mathbb{T} \underset{\sigma}{\overset{\tau}{\rightleftarrows}} \mathbb{U}$$

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

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

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 posses 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 $\mathbb{T}$ to a small cartesian closed category $\mathcal{S}(\mathbb{T})$. This extends to a functor

$$\mathcal{S} : \lambda\mathsf{Thr} \to \mathsf{Ccc} \ ,$$

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

Theorem 2.4.4 *The functors $\mathbb{L} : \mathsf{Ccc} \to \lambda\mathsf{Thr}$ and $\mathcal{S} : \lambda\mathsf{Thr} \to \mathsf{Ccc}$ constitute an equivalence of categories, "up to equivalence". This means that for any $\mathcal{C} \in \mathsf{Ccc}$ there is an equivalence of catgories*

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

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

$$\mathbb{T} \simeq \mathbb{L}(\mathcal{S}(\mathbb{T})) \ .$$

Proof. For a small cartesian closed category $\mathcal{C}$, consider the functor $\eta_{\mathcal{C}} : \mathcal{C} \to \mathcal{S}(\mathbb{L}(\mathcal{C}))$, 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 $A \in \mathcal{C}$,

$$x : \ulcorner A \urcorner \mid \ulcorner 1_A \urcorner x = x : \ulcorner A \urcorner$$

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 $X \in \mathcal{S}(\mathbb{L}(\mathcal{C}))$ 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

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

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 $\theta_{\mathcal{C}} : \mathcal{S}(\mathbb{L}(\mathcal{C})) \to \mathcal{C}$ so that $\theta_{\mathcal{C}} \circ \eta_{\mathcal{C}} \cong 1_{\mathcal{C}}$ and $\eta_{\mathcal{C}} \circ \theta_{\mathcal{C}} \cong 1_{\mathcal{S}(\mathbb{L}(\mathcal{C}))}$.

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

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

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

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

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

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

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

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

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

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

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

1. If $X = 1$ then $\tau_{\mathbb{T}}(\sigma_{\mathbb{T}} 1) = 1$.
2. If $X = \ulcorner A \urcorner$ is a basic type then A is a type in $\mathbb{T}$. We proceed by induction on the structure of A:
   (a) If $A = 1$ then $\tau_{\mathbb{T}}(\sigma_{\mathbb{T}} \ulcorner 1 \urcorner) = 1$. The types 1 and $\ulcorner 1 \urcorner$ are isomorphic via the constant $\mathtt{T} : 1 \to \ulcorner 1 \urcorner$.
   (b) If A is a basic type then $\tau_{\mathbb{T}}(\sigma_{\mathbb{T}} \ulcorner A \urcorner) = \ulcorner A \urcorner$.

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

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

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

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

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

■

Exercise 2.4.5 In the previous proof we defined, for each $\mathcal{C} \in \mathsf{Ccc}$, a functor $\eta_C : \mathcal{C} \to \mathcal{S}(\mathbb{L}(\mathcal{C}))$. Verify that this determines a natural transformation $\eta : 1_{\mathsf{Ccc}} \Longrightarrow \mathcal{S} \circ \mathbb{L}$. Can you say anything about naturality of the translations $\tau_{\mathbb{T}}$ and $\sigma_{\mathbb{T}}$? What would it even mean for a translation to be natural?