Chapter 11

SECOND ORDER LAMBDA CALCULUS

The system λ_2, or second order λ-calculus, has been introduced by Girard for the sake of Proof Theory. It was meant to prove, in particular, a normalization theorem for second order arithmetic, PA_2 (also considered a sound formalization of analysis, by proof theorists). The key points are that (second order) λ-terms code proofs in (second order) systems based on natural deduction and, quite generally, that cut elimination corresponds to β reduction for λ-terms. Thus normal proofs for and consistency of PA_2 follow from the normalization theorem for λ_2 (see chapter 8 and references).

This calculus was later rediscovered by Reynolds and, since then, it has received great attention, mostly within the computer science community, from both the syntactic (typing, consistent extensions, etc.) and semantic points of view. The main novelty of λ_2, over the simply typed calculus, is the possibility of abstracting a term with respect to *type* variables; by this, λ_2 represents "polymorphic" functions, that is, functions that may have several types or, more precisely, that may update their own type.

It is largely agreed that this specific formalization of the broad notion of polymorphism in programming focuses most of the main concerns of the programmers who apply these methods and suggests relevant implementations of modularity in functional programming.

The type system of λ_2 is an extension of the simple types in section 8.2, and it is meant to provide a type for polymorphic functions (i.e., terms obtained by type abstraction). As we said, this is achieved by introducing *type variables* and allowing a quantification (informally, a product) over all types. The type $\forall X{:}Tp.T$ is the type of all those terms that, when applied to a type S, yield a new term of type $[S/X]T$. Types are defined impredicatively since in the inductive definition of the collection Tp of all types one has to use Tp itself, which is being defined. For example, Tp is used when defining $\forall X{:}Tp.T$ as a type or, also, $\forall X{:}Tp.T$ is said to be a type, while it contains a quantification over *all* types, including itself. The "dimensional clash" (and the semantic difficulty) which derives from impredicativity is evident when considering that a term of type $\forall X{:}Tp.T$ can be applied to its own type. This circularity, very common in mathematics (e.g., for least upper bounds and related notions) comes with the expressive power of the system and is reflected in the difficulty of finding sound mathematical models for it. It is our opinion that Internal Category Theory provides significant tools for this semantic investigation. First, it allows the explicit use of "constructive" universes, as an alternative to the (usually intended) set-theoretic frame for (small) categories, where the constructions below would provide trivial models. Second, it reflects and gives meaning to the circularity of impredicativity by a mathematically clear closure property of some internal categories.

11.1 Syntax

Types and (rough) terms are first defined in BN-form. The typing rules will pick up, among the rough terms, the legal ones. Types are built up from type variables, ranged by X, Y, Z…; terms are built up from (term) variables, ranged by x, y,…:

Type expressions: $T := X \mid (T \to S) \mid (\forall X{:}Tp.T\ )$

Term expressions: $e := x \mid (ee) \mid (eT) \mid (\lambda x{:}T.e) \mid (\Lambda X{:}Tp.e)$.

We use capital letters T,S,... as metavariables for type expressions.

Conventions: λ, Λ and $\forall$ are all variable binders. An unbound variable x in e is **free** in e (*notation:* $x \in FV(e)$). The **substitution** of a for x in e ([a/x]e) is defined by induction, provided that a is free for x in e, as usual.

A **context** is a finite set Γ of type variables; ΓX stands for $\Gamma \cup \{X\}$. A type T is **legal** in Γ iff $FV(T) \subseteq \Gamma$. A **type assignment in** Γ is a finite list $E = (x_1{:}T_1),\ldots, (x_n{:}T_n)$ such that any T_i is legal in Γ.

The typing relation $\Gamma;E \vdash e{:}\ T$, where E is a type assignment legal in Γ, e is a term expression and T is a type expression, is defined as follows:

(assumption) $\qquad \Gamma;E \vdash x{:}T \quad$ if $(x{:}T) \in E$

(→I) $$\frac{\Gamma;E(x{:}T) \vdash e{:}\ S}{\Gamma;E \vdash (\lambda x{:}T.e) : (T \to S)}$$

(→E) $$\frac{\Gamma;E \vdash f{:}\ (T \to S) \qquad \Gamma;E \vdash e{:}\ T}{\Gamma;E \vdash (fe) : S}$$

(∀I) $$\frac{\Gamma X;\ E \vdash e : T}{\Gamma;E \vdash (\Lambda X{:}\ Tp.\ e) : (\forall X : Tp.\ T)} \qquad (*)$$

* if there is no variable in FV(e) whose type depends on X

(∀E) $$\frac{\Gamma;E \vdash f : (\forall X : Tp\ .\ T) \qquad \Gamma \vdash S : Tp}{\Gamma;E \vdash (fS) : [S/X]T}$$

Conversion Equations among well-typed terms are defined by the following axioms:

β.	$(\lambda x{:}A\,.b)\, e = [e/x]b$	
β_2.	$(\Lambda X{:}Tp\,.b)\, A = [A/X]b$	
η.	$\lambda x{:}A\,.bx = b$	if $x \notin FV(b)$
η_2.	$\Lambda X{:}Tp\,.bX = b$	if $X \notin FV(b)$

and by the usual rules that turn "=" into a congruence relation.

Before starting the formal definition of the models for such a system, it is worthwhile to say a few words about its interpretation, and to try to explain the following work by a naive presentation of the categorical meaning of λ_2. Note also that this chapter is followed by one entirely dedicated to examples of the abstract categorical treatment which we follow here. The reader may find convenient to check hi/her understanding of it against the structures presented in chapter 12.

As pointed out in chapter 8, any Cartesian closed category **C** can be used to give a categorical semantics to the simply typed lambda calculus: types (which are just constant types or "arrow types") are interpreted by objects of **C**; terms of type T, with free variables $x_1{:}T_1,\ldots, x_n{:}T_n$, are interpreted as morphisms from $T_1\times\ldots\times T_n$ to T. The categorical interpretation of the second order calculus generalizes this semantics; in this case, however, the collection of types Tp must be closed not only under the arrow construction, but also under universal quantification. If we write $\forall X{:}Tp.T$ as $\forall(\underline{\lambda}X{:}Tp.T)$, where $\underline{\lambda}$ is an informal lambda notation for functions, $\forall$ may be readily understood as a map from $(Tp \to Tp)$ to Tp, as it turns the map $\underline{\lambda}X{:}Tp.T$ in $(Tp\to Tp)$ into a type. Thus, the interpretation of $\forall$ should be a map from $Ob_C\to Ob_C$ to Ob_C. The problem is that this map has to be represented *internally* in some ambient category. A natural choice is to have some sort of metacategory E such that **C** may be regarded as an internal category of E, in the sense of section 7.2. Recall, in short, that **C** must consist of a pair (c_0, c_1) of objects of E such that, informally, $Ob_C = c_0$ and $Mor_C = c_1$. If E is Cartesian closed, then $\forall$ may typed as $\forall$: ${c_0}^{c_0}\to c_0$.

Objects of the kind $Tp\to Tp$ (i.e. "points," or "elements" of ${c_0}^{c_0}$) are usually called *variable types*. As we have already seen, if σ is a variable type, the type $\forall(\sigma)$ represents intuitively the collection of all the polymorphic terms e such that, for all types T, $(eT) : \sigma(T)$. This is equivalent to saying that $\forall(\sigma)$ is a *dependent product* that is, a product of different copies of c_0 indexed by σ on elements of c_0 itself. The projections of this dependent product yield the instances of the polymorphic terms in $\forall(\sigma)$ with respect to particular types. In other words, there will be in the model an operation proj: $(c_0\to c_0)\times c_0\to c_1$ that takes a variable type σ: $c_0\to c_0$, a type T, and gives a morphism $proj_\sigma(T)$: $\forall(\sigma)\to\sigma(T)$; $proj_\sigma(T)$ describes how a polymorphic term of type $\forall(\sigma)$ can be instantiated into the type $\sigma(T)$, thus modeling the application of a term e in $\forall(\sigma)$ to a type T.

By the definition of a dependent product, we will also have an isomorphism between the polymorphic terms in $\forall(\sigma)$ and the collection of all the families $\{e_T{:}\ \sigma(T)\}_{T\in c_0}$ of terms indexed over all types. Let us call Δ this isomorphism, which relates a family of terms $\{e_T: \sigma(T)\}_{T\in c0}$ to the polymorphic term $\Delta(\{e_T: \sigma(T)\}_{T\in c0}) = \Lambda T{:}Tp.e_T : \forall(\sigma)$. The functions proj and Δ satisfy the following equations:

1. $proj_\sigma(S)\ (\ \Delta(\{e_T\}_{T\in c_0})\ ) = e_S$;
2. $\Delta(\{proj_\sigma(T)(e)\}_{T\in c_0}) = e$;

whose meaning may be sketched as follows:

1. if we define a polymorphic function from the collection of functions $\{e_T\}_{T\in c0}$, and then we consider the particular instance relative to the type S, this is equal to the original function e_S;
2. if, given a polymorphic function e, we consider the collection of all its instances $\{proj_\sigma(T)(e)\}_{T\in c_0}$ and then we use this collection to rebuild a polymorphic function, we obtain e again.

Equations 1. and 2. above are the key facts allowing the interpretation of rules β and η, respectively, for second order abstraction and application.

Exercise Compare equations 1 and 2 above with the equations for the categorical Cartesian product

$$p_i \circ \langle f_1,f_2\rangle = f_i \quad \text{for } i = 1,2 ;$$
$$\langle p_1 \circ f, p_2 \circ f \rangle = f .$$

11.2 The External Model

The informal discussion of the previous section should have motivated the use of internal concepts in describing the semantics of λ_2. The model definition inspired by these ideas will be the main object of study in this chapter and it is presented in the following section. We introduce here a different notion of categorical model that does not require the use of internal concepts. It is based on an algebraic generalization of the semantics of the simply typed lambda calculus in a bidimensional universe of Cartesian closed categories indexed over another (global) CCC. We will call this model "external." In this model the collection of types is represented by a single object of the global category, say c_0 (or Ω, as it is usually denoted in this approach), but no requirement is made in order to have an internal category with c_0 as an object of objects. This fact, however, must be heavily compensated for by a number of particular conditions that relate "on the nose" categorical properties of indexed categories, which are not very intuitive. Thus, on one hand, the external model is more manageable then the internal one; on the other, it is less limpid and, in a sense, less suggestive. We claim that both these properties of the external model are due to the fact that it is a particularly simple instance of the internal notion. More specifically, we will show that an external model is just an internal one whose ambient category is a topos of presheaves, and whose object of objects c_0 is a representable functor $[\_,\Omega]$. The understanding we propose of the external model also sheds some light on the interplay among the different conditions in its definition and gives a new justification for some apparently ad hoc requirements.

At the base of the external notion of a model for λ_2, there is the notion of a class of small categories indexed over another (global) category E - essentially a contravariant functor G from E to **Cat**. E is a Cartesian category with a distinguished object Ω, which interprets the collection of types. Products Ω^n are used to give meaning to contexts. Arrows in $E[\Omega^n,\Omega]$ represent types with at most n free variables in the context Ω^n.

The functor $G: E\to$**Cat** takes every context Ω^n in E to a (local) category $G(\Omega^n)$ whose objects are the types legal in that context. Thus these types appear both as arrows in E and as objects in the local categories, and it is natural to require $Obj(G(e)) = hom_E(e, \Omega)$. The arrows between two types σ and τ in a local category $G(\Omega^n)$ correspond to terms of type τ and free variables in σ. Every local category is required to be a model of a simply typed lambda calculus and, thus, it is Cartesian closed. As for the interpretation of the polymorphic product it is described by an adjoint situation between local categories; moreover this adjointness must be natural with respect to the global parameter given by the context.

11.2.1 Definition *An **external λ_2 model** (PL category) is a triple (E, G, Ω) where:*

1. E is a Cartesian closed category (global category);

2. Ω is a distinct object in E;

*3. $G: E^{op} \to$ **Cat** is a functor such that*

i. for each object e in E, $Obj(G(e)) = hom_E(e, \Omega)$, and, for each morphism $\sigma\in E[e',e]$, the functor $G(\sigma): G(e) \to G(e')$ acts on the objects of $G(e)$ as $hom_E(\sigma,\Omega)$.

ii. for each object e in E, the (local) category $G(e)$ is Cartesian closed; for every $\sigma\in E[e',e]$, the functor $G(\sigma): G(e) \to G(e')$ preserves the Cartesian closed structure "on the nose" (and not just up to isomorphism); that is, for $a,b\in Obj_{G(e)} = hom_E(e, \Omega)$ it satisfies:

a. $G(\sigma)(t_{G(e)}) = t_{G(e')}$, where $t_{G(e)}$ is the terminal object in $G(e)$

$G(\sigma)(!_a) = !_{G(\sigma)(a)}$

b. $G(\sigma)(a \times_{G(e)} b) = G(\sigma)(a) \times_{G(e')} G(\sigma)(b)$, where $\times_{G(e)}$ is the product in $G(e)$

$G(\sigma)(fst_{a,b}) = fst_{G(\sigma)(a),G(\sigma)(b)}$

$G(\sigma)(snd_{a,b}) = snd_{G(\sigma)(a),G(\sigma)(b)}$

c. $G(\sigma)([a,b]_{G(e)}) = [G(\sigma)(a), G(\sigma)(b)]_{G(e')}$, where $[,]_{G(e)}$ is the exponent in $G(e)$

$G(\sigma)(eval_{a,b}) = eval_{G(\sigma)(a),G(\sigma)(b)}$

*iii. an E-indexed adjunction $<$ **Fst**, $\forall$, $\Delta > : G \to G^\Omega$, where*

*a. $G^\Omega : E^{op} \to$ **Cat** (see definition 7.1.2) is the functor defined by*

$\forall e\in Ob_E \quad G^\Omega(e) = G(e\times\Omega)$

$\forall\sigma\in E[e',e] \quad G^\Omega(\sigma) = G(\sigma\times id_\Omega)$

*b. $\forall e\in Ob_E$, **Fst**$(e) = G(fst_{e,\Omega}) : G(e) \to G^\Omega(e) = G(e\times\Omega)$ (with $fst_{e,\Omega}: e\times\Omega\to e$).*

By definition 7.1.6 of an E-indexed adjunction, we have, for every object e in E, an adjunction

$$< G(fst_{e,\Omega}), \forall(e), \Delta(e) > : G(e) \to G(e\times\Omega)$$

and, moreover, $\Delta(e') \circ G(\sigma\times id_{\Omega}) = G(\sigma) \circ \Delta(e)$

11.2.2 Remark If (E,G) is an external λ_2 model, then we have the following natural transformations:

$$\times_{G(\_)}: \quad hom_{E\times E}(K^2(\_), (\Omega,\Omega)) \to hom_E(\_, \Omega),$$
$$[,]_{G(\_)}: \quad hom_{E\times E}(K^2(\_), (\Omega,\Omega)) \to hom_E(\_, \Omega),$$
$$\forall: \quad hom_E(\_\times\Omega, \Omega) \to hom_E(\_, \Omega),$$

where K^2 is the diagonal functor.

Indeed conditions 3.ii.b and 3.ii.c in definition 11.2.1 express exactly the naturality of $\times_{G(\_)}$ and $[,]_{G(\_)}$, while by definition $\forall$ is natural from G^{Ω} to G and, a fortiori, also from $hom_E(\_\times\Omega, \Omega)$ to $hom_E(\_, \Omega)$.

11.2.3 Lemma *For every object a in E there are morphisms*

$$x_0 : \Omega x\Omega \to \Omega$$
$$[,]_0 : \Omega x\Omega \to \Omega$$
$$\forall_0: \Omega^{\Omega} \to \Omega$$

such that, for each object e of E and for all objects σ,τ of G(e),

$$x_0 \circ <\sigma,\tau> = \sigma \times_{G(e)}\tau$$
$$[,]_0 \circ <\sigma,\tau> = [\sigma,\tau]_{G(e)}$$

and, for each object ρ of $G(e\times\Omega)$,

$$\forall_0 \circ \Lambda(\rho) = \forall(e)(\rho)$$

Proof We have the following natural transformations

$$\varepsilon_1 = \times_{G(\_)} \circ <,>^{-1} : hom_E(\_, \Omega\times\Omega) \to hom_E(\_, \Omega)$$
$$\varepsilon_2 = [,]_{G(\_)} \circ <,>^{-1} : hom_E(\_, \Omega\times\Omega) \to hom_E(\_, \Omega)$$
$$\varepsilon_3 = \forall \circ \Lambda^{-1} : hom_E(\_, \Omega^{\Omega}) \to hom_E(\_, \Omega)$$

where

$$\times_{G(\_)}: \quad hom_{E\times E}(K^2(\_), (\Omega,\Omega)) \to hom_E(\_, \Omega)$$
$$[,]_{G(\_)}: \quad hom_{E\times E}(K^2(\_), (\Omega,\Omega)) \to hom_E(\_, \Omega)$$
$$\forall: \quad hom_E(\_\times\Omega, \Omega) \to hom_E(\_, \Omega)$$

are the natural transformations of remark 11.2.2 and

$$<,>^{-1}: \quad hom_E(\_, \Omega\times\Omega) \to hom_{E\times E}(K^2(\_), (\Omega,\Omega))$$
$$\Lambda^{-1}: \quad hom_E(\_, \Omega^{\Omega}) \to hom_E(\_\times\Omega, \Omega)$$

are the natural isomorphisms given by the Cartesian closure of E.

Then, by the Yoneda lemma, the arrows x_0, $[,]_0$, $\forall_0$ with the requested properties are obtained by setting

$$x_0 = \varepsilon_1(\Omega\times\Omega)(id_{\Omega\times\Omega})$$
$$[,]_0 = \varepsilon_2(\Omega\times\Omega)(id_{\Omega\times\Omega})$$
$$\forall_0 = \varepsilon_3(\Omega^\Omega)(id_{\Omega^\Omega})$$

For example, we have, for σ,τ: $e \to \Omega$,

$$\begin{aligned} x_0 \circ <\sigma,\tau> &= \varepsilon_1(\Omega\times\Omega)(id_{\Omega\times\Omega}) \circ <\sigma,\tau> \\ &= hom_E[<\sigma,\tau>, \Omega\times\Omega]\ (\varepsilon_1(\Omega\times\Omega)(id_{\Omega\times\Omega})) \\ &= \varepsilon_1(e)(hom_E[<\sigma,\tau>, \Omega\times\Omega](id_{\Omega\times\Omega})) \\ &= \varepsilon_1(e)\ <\sigma,\tau> \\ &= \sigma \times_{G(e)}\tau. \end{aligned}$$

The other equations are proved similarly. ♦

11.3 The External Interpretation

In this section we define, in several steps, the "external" interpretation of the second order lambda calculus.

11.3.1 Type Expressions A type expression T legal in a context $\Gamma = \{X_1,\ldots,X_n\}$ is interpreted by a morphism $[T]_\Gamma$: $[\Gamma]\to[A]$ in E (where $[\Gamma] = c_0{}^n=((t\times c_0)\times\ldots\times c_0)$, inductively defined as follows:

1. $[X_i]_\Gamma = snd\circ fst^{n-i}$;
2. $[S\to T]_\Gamma = [,]_0 \circ < [S]_\Gamma, [T]_\Gamma >$;
3. $[\forall X{:}Tp.T]_\Gamma = \forall_0 \circ \Lambda([T]_{\Gamma X})$.

Note that, as in the simply typed λ-calculus, variables are projections and arrows are exponents. Morover, impredicative types are interpreted by formalizing (externally) the informal discussion at the end of 11.2 (see also 11.5.1).

11.3.2 Type assignment A legal type assignment $E = (z_1: S_1)\ldots (z_n: S_n)$, in a context Γ, is interpreted by the product (local in $G([\Gamma])$)

$$[E]_\Gamma = (\ldots(t_{G([\Gamma])} \times[S_1]_\Gamma)\ldots)\times[S_n]_\Gamma = x_o{}^n \circ <\ldots<t_{G([\Gamma])},[S_1]_\Gamma>\ldots,[S_n]_\Gamma >$$

where $t_{G([\Gamma])}$ is the terminal object in the local category $G([\Gamma])$.

11.3.3 Terms A legal term M such that

$$\Gamma = X_1,\ldots,X_n\ ;\ E = (z_1: S_1)\ldots (z_n: S_n)\ \vdash\ M : T$$

is interpreted by a morphism

$[M]_{\Gamma E}$: $[E]_\Gamma \to [T]_\Gamma$ in the local category $G([\Gamma])$.

The inductive definition is

1. $[z_i]_{\Gamma E} = snd \circ fst^{n-i}$;
2. $[MN]_{\Gamma E} = eval \circ <[M]_{\Gamma E}, [N]_{\Gamma E}>$;
3. $[\lambda x:S.M]_{\Gamma E} = \Lambda([M]_{\Gamma E[x:S]})$;
4. $[\Lambda X:Tp.M]_{\Gamma E} = \mathbb{\Delta}([M]_{\Gamma X;E})$;
5. $[M[T]]_{\Gamma E} = G(<id, [T]_\Gamma>) (Proj_{([\Gamma])}) \circ [M]_{\Gamma E}$,

where $Proj_e$ is the counit of the adjunction $< G(fst_{e,\Omega}), \forall(e), \mathbb{\Delta}(e) > : G(e) \to G(e \times \Omega)$, i.e., $Proj_e = \mathbb{\Delta}(e)^{-1}(id)$.

11.3.4 Remark The interpretation we have just given is somewhat informal. Indeed we should always specify in which local category we are working in, and we should add a lot of indices for the natural transformations. Note also that the interpretation is not, as it could seem, by induction on the syntactical structure of terms, but on the lenghth of the proof of their derivation (the proof that the terms are well typed). For example, a fully specified interpretation for $[\Lambda X:Tp.M]_{\Gamma E}$ would be: if $\Gamma X; E \vdash M: T$ and $\tau = [T]_{\Gamma X}$ then $[\Lambda X:Tp.M]_{\Gamma E} = (\Delta([\Gamma])([E]_\Gamma, \tau)) ([M]_{\Gamma X;E})$.

11.4 The Internal Model

In this section we define the notion of internal λ_2 model. The intuition is to require for an internal Cartesian closed category $c \in Cat(E)$ the existence of the arrow $\forall_0: c_0{}^{c0} \to c_0$, which gives the depended product, in a such a way that equations 1 and 2 of section 11.1 are verified. We obtain by this a characterization of internal models by means of *ground* equations, with the consequence that internal models are preserved by limit- and exponent-preserving functors.

In what follows, we always assume that the ambient category E is Cartesian closed and has finite limits.

11.4.1 Definition *Let a be an object of E. |a| is the internal category (a, a, id, id, id, id).*

The internal |a| represents (internalizes) the discrete category with exactly one morphism (the identity) for each point in a.

11.4.2 Definition *Let $c = (c_0, c_1, DOM, COD, COMP, ID) \in Cat(E)$. Define then:*
$\boldsymbol{c*} = (c_0{}^{c0}, c_1{}^{c0}, DOM*, COD*, COMP*, ID*)$ *with*

$DOM* = \Lambda(DOM \circ eval)$

$COD* = \Lambda(COD \circ eval)$

$COMP* = \Lambda(COMP \circ eval \times_0 eval \circ p)$

$ID_* = \Lambda(ID \circ eval)$

where $p = \langle\langle \Pi_1 \circ p_1, p_2 \rangle, \langle \Pi_2 \circ p_1, p_2 \rangle\rangle_0 : (c_1{}^{c0} \times_0 c_1{}^{c0}) \times c_0 \to (c_1{}^{c0} \times c_0) \times_0 (c_1{}^{c0} \times c_0)$.

The idea behind the previous definition is that the object c_* represents internally the category of functors from $|c_0|$ to c. Note that, as $|c_0|$ is a discrete category, the functors F from $|c_0|$ to c are fully determined by their functions f_0 on objects. In effect, this informal idea may be formalized by the following remark: if E is Cartesian closed, so it is $Cat(E)$. The object c_* of the previous definition is isomorphic in $Cat(E)$ to the exponent of $|c_0|$ and c. The category c_* may be also regarded as the collection of all tuples of elements of c indexed by elements in c_0, for $|c_0|$ is a discrete category.

11.4.3 Definition *The constant internal functor* $K : c \to c_*$ *is* $K = (k_0, k_1)$ *with*

$k_0 = \Lambda(fst) : c_0 \to c_0{}^{c0}$ *where* $fst: c_0 \times c_0 \to c_0$ *is the projection*

$k_1 = \Lambda(fst) : c_1 \to c_1{}^{c0}$ *where* $fst: c_1 \times c_0 \to c_1$ *is the projection.*

$K : c \to c_*$ must be considered as a sort of diagonal functor. Informally, given an object b in c_0, its image under K is the tuple (indexed on c_0) of all-b elements (i.e., the constant-b function from c_0 to c_0). As a right adjoint to the diagonal functor $K^2: C \to C^2$ yields the categorical product, similarly, a right adjoint to the functor $K : c \to c_*$ yields the (categorical) dependent product indexed over c_0.

11.4.4 Definition *A* ***model for*** $\boldsymbol{\lambda_2}$ *is given by*

1. a Cartesian closed category E *with all finite limits (global category);*

2. an internal Cartesian closed category $c = (c_0, c_1, DOM, COD, COMP, ID) \in Cat(E)$*;*

3. a right (internal) adjoint to $K: c \to c_*$ *.*

The requirements on the global category E in the previous definition could be slightly relaxed: the notion of internal category can be also given in interesting ambient categories without *all* limits (see the next chapter for an example). Similarly, for a model of λ_2 we actually need only exponents of the form e^{c0} in E. Our requirements are very close to those needed for models of the stronger calculus $F\omega$. In this case the only further condition is that of having a right (internal) adjoint to $K_e: c \to c_e$ for every object e of E, where $c_e = (c_0{}^e, c_1{}^e, DOM_e, COD_e, COMP_e, ID_e)$ represents the category of functors from $|e|$ to c, and $K_e: c \to c_e$ is the internal functor defined by

$$K_e = (k_{e,0} = \Lambda(fst): c_0 \to c_0{}^e,\ k_{e,1} = \Lambda(fst): c_1 \to c_1{}^e).$$

By theorem 7.3.7, a right adjoint to $K: c \to c_*$ is fully determined by

i. an arrow $\forall_0: c_0{}^{c0} \to c_0$,

ii. an arrow PROJ: $c_0{}^{c_0} \to c_1{}^{c_0}$ such that $DOM_* \circ PROJ = k_0 \circ \forall_0$, $COD_* \circ PROJ = id$.
(if $h: e \to c_0{}^{c_0}$, we use the abbreviation $PROJ_h$ for $PROJ \circ h$)

iii. an arrow Δ: X"→ Y" where X" and Y" are respectively the pullbacks of

$< DOM_* , COD_* > : c_1{}^{c_0} \to c_0{}^{c_0} \times c_0{}^{c_0}$, $k_0 \times id : c_0 \times c_0{}^{c_0} \to c_0{}^{c_0} \times c_0{}^{c_0}$

$< DOM,COD > : c_1 \to c_0 \times c_0$, $id \times \forall_0 : c_0 \times c_0{}^{c_0} \to c_0 \times c_0$

$$\begin{tikzcd}
X'' \arrow[r, "\Pi_{X''}"] \arrow[d, "\gamma"] \arrow[rr, bend left, "\Delta"] & c_1{}^{c_0} \arrow[d, "<DOM_*,COD_*>"] & Y'' \arrow[r, "\Pi_{Y''}"] \arrow[d, "\gamma'"] & c_1 \arrow[d, "<DOM,COD>"] \\
c_0 \times c_0{}^{c_0} \arrow[r, "\Lambda(fst)\times id"] & c_0{}^{c_0} \times c_0{}^{c_0} & c_0 \times c_0{}^{c_0} \arrow[r, "id\times\forall_0"] & c_0 \times c_0
\end{tikzcd}$$

such that:

g0. $\gamma' \circ \Delta = \gamma$

g1. $(PROJ \circ p_2 \circ \gamma) * (\Lambda(fst) \circ \Pi_{Y''} \circ \Delta) = \Pi_{X''}$

h. $\Delta \circ < \gamma', (PROJ \circ p_2 \circ \gamma') * (\Lambda(fst) \circ \Pi_{Y''}) >_0 = id_{Y''}$

where $h*k = COMP_* \circ <h,k>_0 = \Lambda(\Lambda^{-1}(h) \circledcirc \Lambda^{-1}(k))$, and $fst: c_1 \times c_0 \to c_1$ is the projection.

PROJ is the counit of the adjunction. In order to understand its meaning, it is useful to compare it with the counit of the Cartesian product. In that case, the counit is

$$(p_{(a1,a2),1}, p_{(a1,a2),2}) \in \mathbf{C}^2[(K^2(a_1 \times a_2), (a_1,a_2)]$$

where K^2 is the diagonal functor. That is, the counit is a collection of morphisms $p_{(a1,a2),i}$ in **C**, indexed over objects (a_1,a_2) of $\mathbf{C}^2$ and objects $i = 1,2$ in the category **2**, such that each $p_{(a1,a2),i}$ has domain $a_1 \times a_2$ and codomain a_i. Analogously, PROJ: $c_0{}^{c_0} \to c_1{}^{c_0} \cong c_0{}^{c_0} \times c_0 \to c_1$ is a collection of morphisms $PROJ_\sigma(T)$ in c_1 indexed over objects σ of c_* and objects T of c (which now corresponds to the category **2** above), such that each projection $PROJ_\sigma(T)$ has domain $\forall_0(\sigma)$ and codomain $\sigma(T)$.

Consider now two points $f: t \to c_1{}^{c_0}$, $g: t \to c_1$ in $c_1{}^{c_0}$ and c_1. Informally, f is a family of terms in c_1 indexed by objects in c_0, and g is a term in c_1. If there exists some a such that $DOM \circ f = K_0 \circ a$, that is if all the terms represented by f have a common domain $a : t \to c_0$, then we can "apply" the isomorphism Δ and obtain the polymorphic term $\Delta \circ f$ (note the usual confusion between application and composition resulting from reasoning about points). Conversely, if there exists b such that $COD \circ g = \forall_0 \circ b$, then g is a polymorphic term of type $\forall_0 \circ b$.

Formally, for every e in E, given $f: e \to c_1{}^{c_0}$ and $g: e \to c_1$ such that

$DOM \circ f = K_0 \circ a$ $\qquad$ $COD \circ f = b$

$DOM \circ g = a$ $\qquad$ $COD \circ g = \forall_{a0} \circ b$

equations (g1) and (h) above give

g1. $\quad PROJ_b * (\Lambda(fst) \circ \Pi_{Y''} \circ \mathbb{\Delta} \circ \langle\langle a,b\rangle,f\rangle_0) = f$

h. $\quad \Delta \circ \langle \langle a,b\rangle, PROJ_b * (\Lambda(fst) \circ g) \rangle_0 = \langle\langle a,b\rangle,g\rangle_0$

and with easy manipulations, recalling that $h*k = COMP* \circ \langle h,k\rangle_0 = \Lambda(\Lambda^{-1}(h) \circledcirc \Lambda^{-1}(k))$, one obtains

g1'. $\quad \Lambda((eval \circ PROJ_b \times id_{c0}) \circledcirc (\Pi_{Y''} \circ \Delta \circ \langle\langle a,b\rangle,f\rangle_0 \circ fst)) = f$

h'. $\quad \Delta \circ \langle \langle a,b\rangle, \Lambda((eval \circ PROJ_b \times id_{c0}) \circledcirc (g \circ fst)) \rangle_0 = \langle\langle a,b\rangle,g\rangle_0$

which are the formalization of equations 1 and 2 of our informal discussion about second order models in section 11.1.

11.5 The Internal Interpretation

Let us now summarize some of the data that come with an internal model. All these objects, morphisms, and functions will be used to give an explicit definition of the interpretation for second order terms as follows:

i. For the global category E:

- a terminal object T, products and exponents
- projections: fst, snd
- evaluation morphism: eval (used for defining COMP$*$)
- pairing function: $\langle,\rangle$
- "currying" function: Λ

ii. For the internal category:

- an arrow $t_0: T \rightarrow c_0$ defining the internal terminal object
- an arrow $x_0 : c_0 \times c_0 \rightarrow c_0$ defining the internal product
- an arrow $[,]_0 : c_0 \times c_0 \rightarrow c_0$ defining the internal exponent
- an arrow $\forall_0: {c_0}^{c0} \rightarrow c_0$ defining the dependent product
- internal projections: FST: $c_0 \times c_0 \rightarrow c_1$, SND: $c_0 \times c_0 \rightarrow c_1$
- internal evaluation morphism: EVAL: $c_0 \times c_0 \rightarrow c_1$
- instantiation morphism: PROJ: ${c_0}^{c0} \rightarrow {c_1}^{c0}$ of the dependent product
- internal pairing: $\langle\!\langle,\rangle\!\rangle$: $(c_0 \times c_0 \times c_0) \times_0 (c_1 \times c_1) \rightarrow (c_0 \times c_0 \times c_0) \times_0 c_1$

  $\langle\!\langle,\rangle\!\rangle(a,b,c, f: a \rightarrow b, g: a \rightarrow c) = (a,b,c, pairing(f,g): a \rightarrow b \times c)$
- internal currying function: $\mathbb{\Lambda}$: $(c_0 \times c_0 \times c_0) \times_0 c_1 \rightarrow (c_0 \times c_0 \times c_0) \times_0 c_1$

  $\mathbb{\Lambda}(a,b,c, f: a \times b \rightarrow c) = (a,b,c, curry(f): a \rightarrow c^b)$
- dependent pairing: $\mathbb{\Delta}$: $(c_0 \times {c_0}^{c0}) \times_0 {c_1}^{c0} \rightarrow (c_0 \times {c_0}^{c0}) \times_0 c_1$

  $\mathbb{\Delta}(a, \sigma, \{e_T: a \rightarrow \sigma(T)\}) = (a, \sigma, dep\_pairing(\{e_T\}): a \rightarrow \forall(\sigma))$.

We point out that an internal model is completely determined by (pullbacks and) a set of *ground* equations, that is, equations without (quantified) free variables; this contrasts with external models (or with standard, “external” Cartesian closed categories, for that matter). An important consequence of this is that *internal models are preserved by limit- and exponent- preserving functors* (that is functors preserving the structure for sources and targets of the data defining the model). This fact will be used later on to relate internal and external models.

Notation For $e \in Ob_E$, $e^n = (((t \times e) \times e) \times \ldots \times e)$ where t is the terminal object of E and e appears n times.

11.5.1 Type Expressions A type expression T legal in a context $\Gamma = \{X_1, ..., X_n\}$ is interpreted by a morphism $[T]_\Gamma : c_0{}^n \to c_0$ in E. In particular:

1. $[X_i]_\Gamma = snd \circ fst^{n-i}$
2. $[S \to T]_\Gamma = [,]_0 \circ < [S]_\Gamma, [T]_\Gamma >$
3. $[\forall X{:}Tp.T]_\Gamma = \forall_0 \circ \Lambda([T]_{\Gamma X})$

11.5.2 Type Assignments A type assignment $E = (z_1: S_1)...(z_m: S_m)$ legal in a context $\Gamma = \{X_1,...,X_n\}$ is interpreted by the product

$$[E]_\Gamma = x_0{}^m \circ <...<t_0 \circ !c_0{}^n, [S_1]_\Gamma>..., [S_m]_\Gamma> : c_0{}^n \to c_0$$

where $x_0{}^1 = x_0$ and, for $i>1$, $x_0{}^i = x_0 \circ (x_0{}^{i-1} \times id)$.

11.5.3.Terms A legal term e such that

$$\Gamma = \{X_1,...,X_n\};\ E = [z_1: S_1]... [z_m: S_m] \vdash e : T$$

is interpreted by a morphism

$$[e]_{\Gamma E} : c_0{}^n \to c_1$$

such that $DOM \circ [e]_{\Gamma E} = [E]_\Gamma : c_0{}^n \to c_0$

$COD \circ [e]_{\Gamma E} = [T]_\Gamma : c_0{}^n \to c_0$

In particular,

1. $[z_i]_{\Gamma E} = SND \circledcirc FST^{n-i}$
   (where for simplicity we omit the “indexes” for FST and SND);
2. if $\Gamma;E \vdash f: S \to T$, $\Gamma;E \vdash e:S$, $\sigma = [S]_\Gamma$, $\tau = [T]_\Gamma$, then
   $[fe]_{\Gamma E} = EVAL_{\sigma,\tau} \circledcirc (\langle\!\langle,\rangle\!\rangle \circ < <<[E]_\Gamma,[S \to T]_\Gamma>, \sigma>_0 , <[f]_{\Gamma E},[e]_{\Gamma E}> >_0);$
3. if $\Gamma;E(x{:}S) \vdash e: T$, $\sigma = [S]_\Gamma$, $\tau = [T]_\Gamma$, then
   $[\lambda x{:}S.e]_{\Gamma E} = \Pi_{y'} \circ \mathbb{\Lambda} \circ < <[E]_\Gamma,\sigma,\tau>, [e]_{\Gamma E(x:S)} >_0 ;$
4. if $\Gamma X; E \vdash e: T$ and $\tau = [T]_{\Gamma X}$, then
   $[\Lambda X{:}Tp.e]_{\Gamma E} = \Pi_{Y''} \circ \mathbb{\Delta} \circ < <[E]_\Gamma,\Lambda(\tau)>, \Lambda([e]_{\Gamma X;E})>_0 ;$

5. if $\Gamma,E \vdash e: \forall X{:}Tp.S$ and $\sigma = \Lambda([S]_{\Gamma X})$, then
$[eT]_{\Gamma E} = \Lambda^{-1}(PROJ_\sigma * (\Lambda(fst) \circ [e]_{\Gamma E})) \circ \langle id, [T]_\Gamma \rangle$
$= (\Lambda^{-1}(PROJ_\sigma) \circledcirc ([e]_{\Gamma E} \circ fst)) \circ \langle id, [T]_\Gamma \rangle$
(where $h * k = COMP* \circ \langle h,k \rangle$; $h \circledcirc k = COMP \circ \langle h,k \rangle$).

Given the above definitions, the proof of a soundness theorem, with the required substitution lemmas, is a routine check (as straightforward as it is tedious and laborious).

11.6 Relating Models

In the previous sections, two different notions of model have been introduced. We are now interested in the relation between them. It will turn out that the two notions are not as distant as they may seem.

We start by an analysis of how we can define an external model from an internal one. The construction is based on the *externalization* process of an internal category via hom-functors presented in chapter 7, which corresponds, essentially, to the Yoneda embedding. Since the hom-functor preserves pullbacks and exponents, we will be able to show that any internal model yields an "equivalent" external one.

Suppose that $c = (c_0,c_1,DOM,COD,COMP,ID) \in Cat(E)$ is an internal model. As the reader has probably imagined, the functor $[\_,c] : E^{op} \to \mathbf{Cat}$ of definition 7.4.2 plays the role of G in the external approach. For ease of reference, we recall here that definition.

11.6.1 Definition *Let $c \in Cat(E)$. The functor $G = [\_,c] : E^{op} \to Cat$ is defined in the following way:*

on objects $e \in E$ $\quad [\_,c] = [e,c]$;

on arrows $\sigma: e' \to e$ $\quad [\_,c](\sigma) = [\sigma,c]$ *is the functor from $[e,c]$ in $[e',c]$ which is defined as $[\sigma,c_0]$ on objects and as $[\sigma,c_1]$ on arrows.*

11.6.2 Lemma *$\forall\, \sigma: e' \to e$, $G(\sigma) = [\sigma,c]: [e,c] \to [e',c]$ acts on the objects of $[e,c]$ (i.e., on $E[e,c_0]$) as $E[\sigma,c_0]$.*

Proof By definition. ♦

11.6.3 Lemma *If c is (internally) Cartesian closed, then, for every e in E, $E_{e,c}$ is Cartesian closed.*

Proof (sketch)

Let 1. $\langle\, O, T, \circledcirc \,\rangle : c \to 1$

2. $\langle\, \Delta, x, \langle\!\langle,\rangle\!\rangle \,\rangle : c \to c{\times}c$

3. $\langle\, x, [,] \;, \mathbb{\Lambda} \,\rangle : c \to c$

be the internal adjunctions given by the Cartesian closure of c. Then $<[\_,F], [\_,G], \Theta > : [\_,c]\to[\_,d]$ is an E-indexed adjunction. By proposition 7.4.11 there are three E-indexed adjunctions

1'. $<[\_,O], [\_,T], o' > : [\_,c]\to[\_,1]$

2'. $<[\_,\Delta], [\_,x], <,>' > : [\_,c]\to[\_,c\times c] \cong [\_,c]\times[\_,c]$

3'. $<[\_,x], [\_, [,]], \Lambda' > : [\_,c]\to[\_,c]$ with parameters in $[\_,c]$.

Hence for every e in E, $[e,c]$ is Cartesian closed, since $[e,1]$ is the terminal category in Cat and $[e,\Delta] : [\_,c]\to[\_,c]\times[\_,c]$ is the diagonal functor. ♦

Propositions 7.4.10 and 7.4.11 allow us to give an explicit definition for the natural isomorphisms in (1')-(3') above. In particular,

given $\sigma,\tau, \gamma : e\to c_0$, and $f: e \to c_1\times c_1$ such that $DOM \circ f = \Delta_0 \circ \sigma$, $COD \circ f = < \tau, \gamma >$

$$<,>'(f) = \Pi_y \circ \langle\!\langle,\rangle\!\rangle \circ (< <\sigma, <\tau, \gamma> >, f >_0) : e \to c_1;$$

given $\sigma,\tau, \gamma : e\to c_0$, and $g: e\to c_1$ such that $DOM \circ g = \sigma$, $COD \circ f = x_0 \circ < \tau, \gamma >$

$$<,>'^{-1}(g) = \Pi_x \circ \langle\!\langle,\rangle\!\rangle^{-1}\circ (< < \sigma, <\tau,\gamma> >, g >_0) : e \to c_1\times c_1;$$

given $\sigma,\tau, \gamma : e\to c_0$ and $f: e\to c_1$ such that $DOM \circ f = x_0 \circ < \sigma,\tau >$, $COD \circ f = \gamma$

$$\Lambda'(f) = \Pi_y' \circ \mathbb{A} \circ (< <<\sigma,\tau>, \gamma >, f) : e \to c_1;$$

given $\sigma,\tau, \gamma : e \to c_0$ and $g: e\to c_1$ such that $DOM \circ g = \sigma$, $COD \circ g = [,]_0 \circ <\tau, \gamma >$

$$\Lambda'^{-1}(g) = \Pi_x' \circ \mathbb{A}^{-1} \circ (< <\sigma,<\tau, \gamma >, g) : e \to c_1.$$

By exercise 7.4.12, given $\sigma,\tau : e\to c_0$, the projections associated to $<,>'$ are derived from the internal projections FST and SND by

$$FST_{\sigma,\tau} = FST \circ <\sigma,\tau> : e\to c_1$$

$$SND_{\sigma,\tau} = SND \circ <\sigma,\tau>: e\to c_1.$$

Note that

$$DOM \circ FST_{\sigma,\tau} = x_0 \circ <\sigma,\tau>$$

$$COD \circ FST_{\sigma,\tau} = \sigma$$

$$DOM \circ SND_{\sigma,\tau} = x_0 \circ <\sigma,\tau>$$

$$COD \circ SND_{\sigma,\tau} = \tau.$$

Analogously, given $\sigma,\tau : e \to c_0$, the counit $EVAL_{\sigma,\tau}$ of Λ' for the object $[,]_0\circ<\sigma,\tau>$ is

$$EVAL_{\sigma,\tau} = EVAL \circ <\sigma,\tau> : e \to c_1$$

where EVAL is the internal evaluation map.

Note that

$$DOM \circ EVAL_{\sigma,\tau} = x_0 \circ <[,]_0\circ <\sigma,\tau>, \sigma >$$

$$COD \circ EVAL_{\sigma,\tau} = \tau .$$

11.6.4 Lemma *Let c be (internally) Cartesian closed. $\forall\sigma: e'\to e$, $[\sigma,c]: [e,c]\to[e,c']$ preserves the Cartesian closed structure "on the nose".*

Proof We only consider the product; the other cases are similar.

$$
\begin{aligned}
\forall\tau,\gamma \text{ in } [e,c] \quad [\sigma,c](\tau\times\gamma) &= [\sigma,c](\ [e,x](\tau,\gamma)\ ) \\
&= [\sigma,c](\ x_0 \circ \langle\tau,\gamma\rangle\ ) && \text{by def. of } [e,x] \\
&= x_0 \circ \langle\tau,\gamma\rangle \circ \sigma \\
&= x_0 \circ \langle\tau \circ \sigma, \gamma \circ \sigma\rangle \\
&= x_0 \circ \langle [\sigma,c](\tau), [\sigma,c](\gamma)\rangle \\
&= [e,x](\ [\sigma,c](\tau), [\sigma,c](\gamma)\ ) \\
&= [\sigma,c](\tau) \times [\sigma,c](\gamma) && \text{by def. of } [e,x]. \ \blacklozenge
\end{aligned}
$$

11.6.5 Lemma *For every e, a objects of E,* $[e,c^{|a|}] \cong [e\times a,c]$.

Proof E is Cartesian closed, thus there are the isomorphisms

$$\Lambda_{e,c_0}: E[e\times a, c_0] \cong E[e, c_0{}^a]$$

$$\Lambda_{e,c_1}: E[e\times a, c_1] \cong E[e, c_1{}^a].$$

Λ_{e,c_0} and Λ_{e,c_1} are respectively the functions on objects and on arrows of a functor $\mathbf{\Lambda}$ from $[e\times a,c]$ to $[e,c^{|a|}]$. Indeed, for every σ: $e\times a\to c_0$

$$
\begin{aligned}
\Lambda_{e,c_1}(id_\sigma) &= \Lambda_{e,c_1}(ID \circ \sigma) \\
&= \Lambda_{e,c_1}(ID \circ eval \circ \Lambda_{e,c_0}(\sigma)\times id\ ) \\
&= \Lambda_{e,c_1}(ID \circ eval) \circ \Lambda_{e,c_0}(\sigma) \\
&= ID_{c^{|a|}} \circ \Lambda_{e,c_0}(\sigma) \\
&= id_{\Lambda(\sigma)}
\end{aligned}
$$

and for every f, g: $e\times a\to c_1$

$$
\begin{aligned}
\Lambda_{e,c_1}(g \circledcirc f) &= \Lambda_{e,c_1}(COMP \circ \langle g,f\rangle) \\
&= \Lambda(COMP \circ \langle eval \circ \Lambda_{e,c_1}(g)\times id, eval \circ \Lambda_{e,c_1}(f)\times id\rangle\ ) \\
&= \Lambda(COMP\circ eval\times eval \circ \langle\Lambda_{e,c_1}(g)\times id, \Lambda_{e,c_1}(f)\times id\rangle\ ) \\
&= \Lambda(COMP\circ eval\times eval \circ p \circ \langle\Lambda_{e,c_1}(g), \Lambda_{e,c_1}(f)\rangle\times id) \\
&= \Lambda(COMP\circ eval\times eval\circ p) \circ \langle\Lambda_{e,c_1}(g), \Lambda_{e,c_1}(f)\rangle \\
&= COMP_{c^{|a|}} \circ \langle\Lambda_{e,c_1}(g), \Lambda_{e,c_1}(f)\rangle \\
&= \Lambda_{e,c_1}(g) \circledcirc \Lambda_{e,c_1}(f)
\end{aligned}
$$

Similarly, $\Lambda_{e,c_0}{}^{-1}$ and $\Lambda_{e,c_1}{}^{-1}$ define the functions on objects and on arrows of $\mathbf{\Lambda}^{-1}$, respectively. ♦

11.6.6 Lemma *Let K:* $c\to c*$ *be the functor of definition 11.4.3. For every e in E,* $\mathbf{\Lambda}^{-1} \circ [e,K] = G(fst) = [fst,c] : [e,c]\to[e,c^{|a|}]$.

Proof On objects σ: $e\to c_0$

$$
\begin{aligned}
(\Lambda^{-1} \circ [e,K])(\sigma) &= \Lambda^{-1}([e,K](\sigma)\ ) \\
&= \Lambda^{-1}(k_0 \circ \sigma\ ) && \text{by def. of } [e,K] \\
&= \Lambda^{-1}(k_0) \circ \sigma\times id
\end{aligned}
$$

$$
\begin{aligned}
&= \mathrm{fst} \circ \sigma\times \mathrm{id} && \text{by def. of } k_0\\
&= \sigma \circ \mathrm{fst}\\
&= [\mathrm{fst},c](\sigma) && \text{by def. of } [\mathrm{fst},c].
\end{aligned}
$$

On arrows $f: e\to c_1$

$$
\begin{aligned}
(\Lambda^{-1} \circ [e,K])(f) &= \Lambda^{-1}([e,K](\ f\ ))\\
&= \Lambda^{-1}(k_1 \circ f\ ) && \text{by def. of } [e,K]\\
&= \Lambda^{-1}(k_1) \circ f \times \mathrm{id}\\
&= \mathrm{fst} \circ f \times \mathrm{id} && \text{by def. of } k_1\\
&= f \circ \mathrm{fst}\\
&= [\mathrm{fst},c](\ f\ ) && \text{by def. of } [\mathrm{fst},c]. \ \blacklozenge
\end{aligned}
$$

11.6.7 Corollary *Let* $\Omega = c_0$ *. Then for every* e *in* E, $<[\mathrm{fst},c],\ [e,\forall] \circ \mathbf{\Lambda},\ \Delta' \circ \Lambda_{e,c_1}> :$ $[e,c]\to[e\times\Omega,c]$ *is an adjunction.*

Proof By lemma 11.6.6, $[\mathrm{fst},c] = \mathbf{\Lambda}^{-1} \circ [e,K]$. Then we have the isomorphisms

$$[e\times\Omega,c][\Lambda_{e,c_0}{}^{-1}([e,K](\sigma)), \tau] \overset{\Lambda_{e,c_1}}{\cong} [e,c^*][\ [e,K](\sigma), \Lambda_{e,c_0}(\tau)] \overset{\Delta'}{\cong} [e,c][\ \sigma, [e,\forall](\Lambda_{e,c_0}(\tau))]. \ \blacklozenge$$

Note that, given $\sigma: e\to c_0$,$\tau: e\times a\to c_0$, and $f: e\times a\to c_1$ such that

$$\mathrm{DOM} \circ f = \Lambda^{-1}([e,K](\sigma)\ ) = \sigma \circ \mathrm{fst}$$
$$\mathrm{COD} \circ f = \tau,$$

we have $\Delta' \circ \Lambda_{e,c_1}(f) = \Pi_Y'' \circ \Delta \circ (<<\sigma, \tau >, \Lambda(f) >_0) : e\to c_1$, where Π_Y'' is as in the diagram after definition 11.4.4.

Analogously, given $\sigma: e\to c_0$, $\tau: e\times a\to c_0$, and $g: e\to c_1$ such that

$$\mathrm{DOM} \circ g = \sigma$$
$$\mathrm{COD} \circ f = [e,\forall](\Lambda(\tau)\ ) = \forall_0 \circ \Lambda(\tau)$$

we have the following:

$$
\begin{aligned}
&(\Delta' \circ \Lambda_{e,c_1})^{-1}(g) =\\
&\quad = \Lambda_{e,c_1}{}^{-1} (\Delta'^{-1}(g))\\
&\quad = \Lambda^{-1}(\Pi_X'' \circ \Delta'^{-1} \circ (<<\sigma, \tau >, g >_0)\ )\\
&\quad = \mathrm{eval} \circ (\Pi_X'' \circ \Delta'^{-1} \circ (<<\sigma, \tau >, g >_0\ )\ )\times \mathrm{id}:\ e\times a\to c_1.
\end{aligned}
$$

In particular, given $\sigma : e \to c_0{}^{c_0}$, the counit $(\Delta' \circ \Lambda_{e,c_1})^{-1}(\mathrm{id}_{[e,\forall](\Lambda(\tau))})$ is

$$
\begin{aligned}
\mathrm{Proj}_{\sigma,\tau} &= \mathrm{eval} \circ (\Pi_X'' \circ \Delta'^{-1} \circ (< <\sigma, \tau >, \mathrm{ID} \circ \forall_0 \circ \Lambda(\sigma) >_0)\ )\times \mathrm{id}:\ e\times c_0 \to c_1\\
&= \mathrm{eval} \circ (\mathrm{PROJ} \circ <s,t>)\times \mathrm{id}\\
&= \Lambda^{-1}(\ (\mathrm{PROJ} \circ <s,t>)
\end{aligned}
$$

where PROJ is the internal counit.

11.6.8 Lemma *The isomorphism of the adjunction in corollary 11.6.7 is also natural in e; that is, for every* $\gamma: e \to e'$, $[\gamma,c] \circ (\Delta' \circ \Lambda_{e,c_1}) = (\Delta' \circ \Lambda_{e,c_1}) \circ [\gamma\times id,c]$.

Proof For every $\gamma: e\to e'$, and $f: e\times a \to c_1$ such that

$$DOM \circ f = \Lambda^{-1}([e,K](\sigma)) = \sigma \circ fst$$

$$COD \circ f = \tau \quad (\text{ where } \sigma: e \to c_0, \tau: e\times a \to c_0)$$

$$\begin{aligned}
([\gamma,c] \circ (\Delta' \circ \Lambda_{e,c_1}))(f) &= [\gamma,c] (\Delta'(\Lambda_{e,c_1}(f))) \\
&= [\gamma,c] (\Pi_y'' \circ \Delta \circ <<\sigma, \tau >, \Lambda(f) >_0) \\
&= \Pi_Y'' \circ \Delta \circ <<\sigma, \tau >, \Lambda(f) >_0 \circ \gamma \\
&= \Pi_Y'' \circ \Delta \circ <<\sigma \circ \gamma , \tau \circ \gamma >, \Lambda(f)\circ \gamma >_0 \\
&= \Pi_Y'' \circ \Delta \circ <<\sigma \circ \gamma , \tau \circ \gamma >, \Lambda(f \circ \gamma\times id) >_0 \\
&= (\Delta' \circ \Lambda_{e,c_1}) (f \circ \gamma\times id) \\
&= ((\Delta' \circ \Lambda_{e,c_1}) ([\gamma\times id,c](f)) \\
&= ((\Delta' \circ \Lambda_{e,c_1}) \circ [\gamma\times id,c]) (f). ♦
\end{aligned}$$

11.6.9 Theorem *If* (E,c) *is an internal* λ_2*-model, then* $(E, c_0, G=[\_,c])$ *is an external* λ_2*-model. Moreover, for any legal expression* Q *of* λ_2*, the internal interpretation of* Q *in* (E,c) *coincides with the external interpretation of* Q *in* $(E, c_0, G=[\_,c])$*; t hat is, they are the same arrow in* E.

Proof Easy, by the previous lemmas. ♦

Now we prove that, using the “internalization” technique of chapter 7, we obtain from any external model $G:E^{op}\to$**Cat** an internal model in the topos of presheaves $E^{op}\to$**Set**. The translation shows that, essentially, any PL-category is nothing else but an internal category in the category of presheaves having as object of objects the contravariant hom-functor. Recall (see definition 7.5.1) that given an E-indexed category $G: E^{op}\to$**Cat** **,** we can build an internal category $\underline{G} = (\underline{G}_0, \underline{G}_1, \underline{DOM}, \underline{COD}, \underline{COMP}, \underline{ID})\in Cat(E^{op}\to$**Set**$)$ in the following way:

for all objects e, e' and arrows $f: e'\to e$ in E:

- $\underline{G}_0: E^{op}\to$**Set** is the functor defined by

$$\underline{G}_0(e) = Ob_{G(e)}$$

$$\underline{G}_0(f) = G(f)_{ob} : Ob_{G(e)}\to Ob_{G(e')}$$

- $\underline{G}_1: E^{op}\to$**Set** is the functor defined by

$$\underline{G}_1(e) = Mor_{A(e)}$$

$$\underline{G}_1(f) = G(f)_{mor} : Mor_{G(e)}\to Mor_{G(e')}$$

- $\underline{DOM}: G_1\to G_0$ is the natural transformation whose components are the domain maps in the local categories, i.e., for $e\in Ob_E$, $\underline{DOM}_e: Mor_{G(e)}\to Ob_{G(e)}$ is defined by

$$\underline{DOM}_e(h:\sigma\to\tau) = \sigma.$$

- $\underline{COD}$, $\underline{ID}$ and $\underline{COMP}$ are defined analogously, “fiber-wise.”

Note in particular that if (E, c_0, G) is a PL category, then $G_0 = E[\_, \Omega]$.

11.6.10 Proposition *If (E, c_0, G) is a PL category, then $\underline{G}$ is an internal Cartesian closed category.*
Proof By proposition 7.5.4. ♦

Before showing that $\underline{G}$ also has an internal dependent product, it is useful to take a closer look at the structure of the involved exponents in $E^{op} \to \mathbf{Set}$.

11.6.11 Lemma *Let $H: E^{op} \to \mathbf{Set}$ be any functor, and let $G_0 = E[\_, \Omega]$. Then their exponent $H^{G_0}: E^{op} \to \mathbf{Set}$ is given, up to isomorphisms, by the following data:*

a. $H^{G_0}(e) = H(e\times\Omega)$
$H^{G_0}(f) = H(f\times id_\Omega)$;

b. $eval : H^{G_0}\times G_0 \to H$
$eval_e(m,f) = H(<id_e, f>)(m)$, *for* $e\in Ob_E$, $m\in H(e\times\Omega)$, $f\in E[e,\Omega]$;

c. $\Lambda : Nat[F\times G_0, H] \cong Nat[F, H^{G_0}]$
$\Lambda(\tau)(e)(m) = \tau_{e\times\Omega}(F(fst)(m), snd)$,
where $\tau: F\times G_0 \to H$, $e\in Ob_E$, $m\in F(e)$, $fst: e\times\Omega \to e$, $snd: e\times\Omega \to \Omega$.

Proof We use the usual definition of exponents in the category of presheaves (see section 3.5) and prove that the one given above is equivalent up to isomorphisms. Remember that

$$H^F(e) = Nat[E[\_,e]\times F, H]$$
$$H^F(f:e'\to e)(\sigma) = \sigma \circ E[\_,f]\times id_F$$

where $E[\_,f]$ is the natural transformation from $E[\_,e']$ into $E[\_,e]$ defined by $E[\_,f] = f \circ \_$. When $F = G_0 = E[\_, \Omega]$, we can use Yoneda's lemma and have

$$H^{G_0}(e) = Nat[E[\_,e]\times E[\_, \Omega], H] \cong Nat[E[\_,e\times\Omega], H] \cong H(e\times\Omega).$$

Let now $f\in E[e',e]$:

$$H^{G_0}(f)(\sigma) = \sigma \circ E[\_,f]\times E[\_,id_\Omega] \cong \sigma \circ E[\_,f\times id_\Omega]\in Nat[E[\_,e'\times\Omega], H] \cong H(e'\times\Omega)$$

Hence, the Yoneda isomorphism yields $H^{G_0}(f) \cong H(f\times id_\Omega)$.

Let us check that the above expressions for eval and Λ satisfy the equations for the exponents. We have to prove that $eval \circ \Lambda(\tau)\times id = \tau$ and $\Lambda(eval \circ h\times id) = h$; let $m\in F(e)$ and $f\in c_0(e)$:

$$
\begin{aligned}
& eval_e((\Lambda(\tau)_e\times id_e)(m,f)) = \\
&= eval_e(\tau_{e\times\Omega}(F(fst)(m), snd), f) && \text{by def. of } \Lambda \\
&= H(<id_e, f>)(\tau_{e\times\Omega}(F(fst)(m), snd)) && \text{by def. of eval} \\
&= \tau_e(F(<id_e, f>)(F(fst)(m)),\ snd \circ <id_e, f>), && \text{by naturality of } \tau \\
&= \tau_e(F(fst \circ <id_e, f>)(m), f) && \text{for F functor} \\
&= \tau_e(m, f)
\end{aligned}
$$

$\Lambda(\mathrm{eval} \circ \mathrm{h}\times\mathrm{id})_e\,(m) =$

$= (\mathrm{eval}_{e\times\Omega} \circ h_{e\times\Omega}\times\mathrm{id}_{e\times\Omega})\,(F(\mathrm{fst})(m), \mathrm{snd})$	by def. of Λ
$= \mathrm{eval}_{e\times\Omega}(\, h_{e\times\Omega}(F(\mathrm{fst})(m)), \mathrm{snd}))$	
$= H(<\mathrm{id}_{e\times\Omega}, \mathrm{snd}>)(\, h_{e\times\Omega}(F(\mathrm{fst})(m))\,)$	by def. of eval
$= H(<\mathrm{id}_{e\times\Omega}, \mathrm{snd}>)(H(\mathrm{fst}\times\mathrm{id}_\Omega)(h_e(m)))$	by naturality of h
$= H(\, \mathrm{fst}\times\mathrm{id}_\Omega \circ\ <\mathrm{id}_{e\times\Omega}, \mathrm{snd}>)(\, h_e(m)\,)$	for G functor
$= H(<\mathrm{fst},\mathrm{snd}>)\,(\, h_e(m)\,)$	
$= h_e(m)$. ♦	

The following lemma exploits the results above in order to give an explicit definition of the constant internal functor $K: \underline{G} \to \underline{G}*$, whose right adjoint will give the depended product:

11.6.12 Lemma *The internal functor* $K = (k_0, k_1): \underline{G} \to \underline{G}*$ *of definition 11.4.3 is given in* $E^{op}\to$***Set*** *by*

$k_0(e) = G(fst)_{obj}$

$k_1(e) = G(fst)_{mor}$

where $fst: e\times\Omega\to e$ *in E.*

Proof Definition 11.4.3 gives the following for K

$k_0 = \Lambda(\mathrm{fst}) : G_0\to G_0{}^{G_0}$ where $\mathrm{fst}: G_0\times G_0\to G_0$ in $E^{op}\to$**Set**

$k_1 = \Lambda(\mathrm{fst}) : G_1\to G_1{}^{G_0}$ where $\mathrm{fst}: G_1\times G_0\to G_1$ in $E^{op}\to$**Set**.

Note first that, for $e\in\mathrm{Ob}_E$, the components of the natural transformations fst above behave as the first projections. Now let $h\in c_0(e)$; then

$$
\begin{aligned}
k_0(e)(h) &= \Lambda(\mathrm{fst})(e)(h)\\
&= \mathrm{fst}_{e\times\Omega}(\, G_0(\mathrm{fst})(h), \mathrm{snd}\,) \quad \text{by lemma 11.6.12, where fst:} e\times\Omega\to e \text{ in E}\\
&= G_0(\mathrm{fst})(h)\\
&= G(\mathrm{fst})(h).
\end{aligned}
$$

Analogously, for any $g\in c_1(e)$:

$$
\begin{aligned}
k_1(e)(g) &= \Lambda(\mathrm{fst})(e)(g)\\
&= \mathrm{fst}_{e\times\Omega}(\, c_1(\mathrm{fst})(g), \mathrm{snd}\,)\\
&= G_1(\mathrm{fst})(g)\\
&= G(\mathrm{fst})(g). \ \blacklozenge
\end{aligned}
$$

11.6.13 Theorem *Let* (E, G, Ω) *be an external model; then* $\underline{G}$ *is an internal model. Moreover, for any legal expression* Q *of* λ_2, *the external interpretation of* Q *in* (E, G, Ω) *coincides with the internal interpretation of* Q *in* $\underline{G}$ *; that is, they are the same arrow in E.*

Proof $\underline{G}$ is Cartesian closed, by proposition 11.6.11. By definition of an external model, the functor G(fst) has a right adjoint $\forall:G^{\Omega}\to G$. In view of lemma 11.6.12, this is all we need for the proof.♦

By the previous theorem, and by the particularly simple way the category $\underline{G}$ is defined from the indexed category (E, G, Ω), every external model can be thought of as an internal model. We could even say that external models *are* the internal categories in the topos of presheaves that have the (contravariant) hom-functor as object of objects (and that have the required internal structure, of course). In this sense, *external models are less general than internal ones*, since they result from fixing some data in an internal model. Note that we have also obtained a posteriori a justification of the apparent simplicity of the external model. This is due to the choice of the well-known topos of presheaves as ambient category and of the hom-functor as canonical object of objects for the internal categories in this topos. This approach, though not fully general, allows a great simplification in the definitions of the involved exponents.

A final comparison between the two approaches is suggetsted by the following remark. Note first that any internal model in a presheaves topos “is” an indexed category; thus, one can think as well of a definition of indexed category model in which also the indexing functor is not representable. On the other hand, if the indexing functor is chosen to be representable, as in the external model, one may wonder why only the object of objects should enjoy this privileged condition. Note that if we suppose that also the object of morphisms is representable, i.e., $c_1 = E[\_, \Omega_1]$, then by the Yoneda embedding, we have an internal model $c = (\Omega, \Omega_1, ...)$ in E.

References The polymorphic lambda calculus was defined in Girard (1971) in his investigation of foundational problems in mathematics. Three years later it was reinvented by Reynolds (1974), who was mainly interested in the type structure of programming languages, testifying the relevance of this formalism for computer science. References to protoype programming languages, where polymorphism is formalized in terms of second order λ-calculi, and a recent application may be found in Cardelli and Longo (1990).

The model definition based on the internal approach is due to Moggi (1985). Unfortunately, since at that time there was no known concrete model that could be described “internally,” his idea was never published, and for some years it remained known only to a restricted number of specialists and collaborators (see Hyland (1987)). Meanwhile a different and in a sense simpler notion of model based on indexed categories was proposed in Seely (1987). Both models are based on the idea in Lawvere (1970) of expressing logical quantifications by means of categorical adjunctions. Further discussions on categorical models of λ_2 may be found in Reynolds (1984), Bainbridge et al. (1987), Hyland and Pitts (1987), Pitts (1987), Longo and Moggi (1988), Scedrov (1988), Reynolds and Plotkin (1988) and Meseguer (1988), among others.

This chapter is derived largely from Asperti and Martini (1989).