Logical Methods in Computer Science
Vol. 7 (3:18) 2011, pp. 1–25
www.lmcs-online.org

Submitted Apr. 4, 2011
Published Sep. 27, 2011

KRIPKE SEMANTICS FOR MARTIN-LÖF'S EXTENSIONAL TYPE THEORY *

STEVE AWODEY [a] AND FLORIAN RABE [b]

[a] Carnegie Mellon University, Pittsburgh, USA
e-mail address: awodey@cmu.edu

[b] Jacobs University Bremen, Germany
e-mail address: florian.rabe@gmail.com

Abstract. It is well-known that simple type theory is complete with respect to non-standard set-valued models. Completeness for standard models only holds with respect to certain extended classes of models, e.g., the class of cartesian closed categories. Similarly, dependent type theory is complete for locally cartesian closed categories. However, it is usually difficult to establish the coherence of interpretations of dependent type theory, i.e., to show that the interpretations of equal expressions are indeed equal. Several classes of models have been used to remedy this problem.

We contribute to this investigation by giving a semantics that is standard, coherent, and sufficiently general for completeness while remaining relatively easy to compute with. Our models interpret types of Martin-Löf's extensional dependent type theory as sets indexed over posets or, equivalently, as fibrations over posets. This semantics can be seen as a generalization to dependent type theory of the interpretation of intuitionistic first-order logic in Kripke models. This yields a simple coherent model theory, with respect to which simple and dependent type theory are sound and complete.

1. Introduction and Related Work

Martin-Löf's extensional type theory ([ML84], MLTT), is a dependent type theory. Its main characteristic is that there are type-valued function symbols that take terms as input and return types as output. This is enriched with further type constructors such as dependent sum and product. The syntax of dependent type theory is significantly more complex than that of simple type theory because well-formed types and terms and their equalities must be defined in a single joint induction.

The semantics of MLTT is similarly complicated. In [See84], the connection between MLTT and locally cartesian closed (LCC) categories was first established. LCC categories interpret contexts Γ as objects $[\![\Gamma]\!]$, types in context Γ as objects in the slice category over

1998 ACM Subject Classification: F.4.1.

Key words and phrases: Kripke models, semantics, type theory, dependent types.

* A preliminary version of this paper appeared as [AR09].

[b] The second author was partially supported by a fellowship for Ph.D. research of the German Academic Exchange Service.

DOI:10.2168/LMCS-7 (3:18) 2011

$[\![\Gamma]\!]$, substitution as pullback, and dependent sum and product as left and right adjoint to pullback. But there is a difficulty, namely that these three operations are not independent: Substitution of terms into types is associative and commutes with sum and product formation, which is not necessarily the case for the choices of pullbacks and their adjoints. This is known as the coherence or strictness problem and has been studied extensively. In incoherent models such as in [Cur89], equal types are interpreted as isomorphic but not necessarily equal objects. In [Car86], coherent models for MLTT are given using categories with attributes. And in [Hof94], a category with attributes is constructed for every LCC category. Several other model classes and their coherence properties have been studied in, e.g., [Str91] and [Jac90, Jac99]. In [Pit00], an overview is given.

These model classes all have in common that they are rather abstract and have a more complicated structure than general LCC categories. It is clearly desirable to have simpler, more concrete models. But it is a hard problem to equip a given LCC category with choices for pullbacks and adjoints that are both natural and coherent. Our motivation is to find a simple concrete class of LCC categories for which such a choice can be made, and which is still general enough to be complete for MLTT.

Mathematically, our main results can be summarized very simply: Using a theorem from topos theory, it can be shown that MLTT is complete with respect to — not necessarily coherent — models in the LCC categories of the form $\mathcal{SET}^P$ for posets P, where $\mathcal{SET}$ is the category of sets and mappings. This is equivalent to using presheaves on posets as models, which are often called Kripke models. They were also studied in [Hof97]. For these rather simple models, a solution to the coherence problem can be given. $\mathcal{SET}$ can be equipped with a coherent choice of pullback functors, and hence the categories $\mathcal{SET}^P$ can be as well. Deviating subtly from the well-known constructions, we can also make coherent choices for the required adjoints to pullback. Finally, rather than working in the various slices $\mathcal{SET}^P/A$, we use the isomorphism $\mathcal{SET}^P/A \cong \mathcal{SET}^{\int_P A}$, where $\int_P A$ is the category of elements: Thus we can formulate the semantics of dependent types uniformly in terms of the simple categories of indexed sets $\mathcal{SET}^Q$ for various posets Q.

In addition to being easy to work with, this has the virtue of capturing the idea that a dependent type S in context Γ is in some sense a type-valued function on Γ: Our models interpret Γ as a poset $[\![\Gamma]\!]$ and S as an indexed set $[\![\Gamma|S]\!] : [\![\Gamma]\!] \to \mathcal{SET}$. We speak of Kripke models because these models are a natural extension of the well-known Kripke models for intuitionistic first-order logic ([Kri65]). Such models are based on a poset P of worlds, and the universe is given as a P-indexed set (possibly equipped with P-indexed structure). This can be seen as the special case of our semantics when there is only one type.

In fact, our results are also interesting in the special case of simple type theory ([Chu40]). Contrary to Henkin models ([Hen50, MS89]), and the models given in [MM91], which like ours use indexed sets on posets, our models are standard: The interpretation $[\![\Gamma|S \to S']\!]$ of the function type is the exponential of $[\![\Gamma|S]\!]$ and $[\![\Gamma|S']\!]$. And contrary to the models in [Fri75, Sim95], our completeness result holds for theories with more than only base types and terms.

A different notion of Kripke-models for dependent type theory is given in [Lip92], which is related to [All87]. There, the MLTT types are translated into predicates in an untyped first-order language. The first-order language is then interpreted in a Kripke-model, i.e., there is one indexed universe of which all types are subsets. Such models correspond roughly to non-standard set-theoretical models.

Signatures	Σ	::=	$\cdot \mid \Sigma, c\!:\!S \mid \Sigma, a\!:\!(\Gamma)\mathtt{type}$
Contexts	Γ	::=	$\cdot \mid \Gamma, x\!:\!S$
Substitutions	γ	::=	$\cdot \mid \gamma, x/s$
Types	S	::=	$a\ \gamma \mid 1 \mid \mathit{Id}(s, s') \mid \Sigma_{x:S}\, S' \mid \Pi_{x:S}\, S'$
Terms	s	::=	$c \mid x \mid *\ \mid \mathit{refl}(s) \mid \langle s, s'\rangle \mid \pi_1(s) \mid \pi_2(s) \mid \lambda_{x:S}\, s \mid s\ s'$

Figure 1: Basic Grammar

We give the syntax of MLTT in Sect. 2 and some categorical preliminaries in Sect. 3. Then we derive the coherent functor choices in Sect. 4 and use them to define the interpretation in Sect. 5. We give our main results regarding the interpretation of substitution, soundness, and completeness in Sect. 6, 7, and 8.

2. Syntax

2.1. **Grammar.** The basic syntax for MLTT expressions is given by the grammar in Fig. 1. The vocabulary of the syntax is declared in signatures and contexts: Signatures Σ declare globally accessible names c for constants of type S and names a for type-valued constants with a list Γ of argument types. Contexts Γ locally declare typed variables x.

Substitutions γ translate from a context Γ to Γ' by providing terms in context Γ' for the variables in Γ. Thus, a substitution from Γ to Γ' can be applied to expressions in context Γ and yields expressions in context Γ'. Relative to a signature Σ and a context Γ, there are two syntactical classes: types and typed terms.

The base types are the application $a\ \gamma$ of a type-valued constant to a list of argument terms γ (which we write as a substitution for simplicity). The composed types are the unit type 1, the identity types $\mathit{Id}(s, s')$, the dependent product types $\Sigma_{x:S}\, T$, and the dependent function types $\Pi_{x:S}\, T$. Terms are constants c, variables x, the element $*$ of the unit type, the element $\mathit{refl}(s)$ of the type $\mathit{Id}(s, s)$, pairs $\langle s, s'\rangle$, projections $\pi_1(s)$ and $\pi_2(s)$, λ-abstractions $\lambda_{x:S}\, s$, and function applications $s\ s'$. We do not need equality axioms $s \equiv s'$ because they can be given as constants of type $\mathit{Id}(s, s')$. For simplicity, we omit equality axioms for types.

Our formulation of MLTT only uses types and terms. This is different from variants of dependent type theory with kinded type families as in [Bar92] and [HHP93]. In particular, in our formulation, the constants a are the only type families, and a itself is not a well-formed expression. All our results extend to the case with kinded type families (see [Rab08]).

Definition 2.1 (Substitution Application)**.** The *application* of a substitution γ to a term, type, or substitution is defined as follows where γ^x abbreviates $\gamma, x/x$.

Substitution in terms:

$$
\begin{array}{lll}
\gamma(c) & := c & \\
\gamma(x) & := s & \text{for } x/s \text{ in } \gamma \\
\gamma(*) & := * & \\
\gamma(\mathit{refl}(s)) & := \mathit{refl}(\gamma(s)) & \\
\gamma(\langle s, s'\rangle) & := \langle \gamma(s), \gamma(s')\rangle & \\
\gamma(\pi_1(s)) & := \pi_1(\gamma(s)) & \\
\gamma(\pi_2(s)) & := \pi_2(\gamma(s)) & \\
\gamma(\lambda_{x:S}\, t) & := \lambda_{x:\gamma(S)}\, \gamma^x(t) & \\
\gamma(f\ s) & := \gamma(f)\ \gamma(s) &
\end{array}
$$

Judgment	Intuition
$\vdash \Sigma\ \mathtt{Sig}$	Σ is a well-formed signature
$\vdash_\Sigma \Gamma\ \mathtt{Ctx}$	Γ is a well-formed context over Σ
$\vdash_\Sigma \gamma : \Gamma \to \Gamma'$	γ is a well-formed substitution over Σ from Γ to Γ'
$\Gamma \vdash_\Sigma S : \mathtt{type}$	S is a well-formed type over Σ and Γ
$\Gamma \vdash_\Sigma S \equiv S'$	types S and S' are equal over Σ and Γ
$\Gamma \vdash_\Sigma s : S$	term s is well-formed with type S over Σ and Γ
$\Gamma \vdash_\Sigma s \equiv s'$	terms s and s' are equal over Σ and Γ

Figure 2: Judgments

Substitution in types:

$$\begin{array}{ll} \gamma(1) & := 1 \\ \gamma(Id(s,s')) & := Id(\gamma(s),\gamma(s')) \\ \gamma(\Sigma_{x:S}\, T) & := \Sigma_{x:\gamma(S)}\, \gamma^x(T) \\ \gamma(\Pi_{x:S}\, T) & := \Pi_{x:\gamma(S)}\, \gamma^x(T) \\ \gamma(a\ \gamma_0) & := a\ \gamma(\gamma_0) \end{array}$$

Substitution in substitutions:

$$\begin{array}{ll} \gamma(\cdot) & := \cdot \\ \gamma(x_1/s_1,\ldots,x_n/s_n) & := x_1/\gamma(s_1),\ldots,x_n/\gamma(s_n) \end{array}$$

Substitution in substitutions is the same as composition of substitutions, and we write $\gamma \circ \delta$ instead of $\gamma(\delta)$.

2.2. **Type System.** The judgments defining well-formed syntax are listed in Fig. 2. The typing rules for these judgments are well-known. Our formulation follows roughly [See84], including the use of extensional identity types. The latter means that the equality judgment for the terms s and s' holds iff the type $Id(s,s')$ is inhabited.

Example 2.2. The theory *Cat* of categories is given by declaring type-valued constants *Ob* and *Mor* and term-valued constants *id* and *comp* such that the following judgments hold

$$\begin{array}{lll} \cdot & \vdash_{Cat}\ Ob & :\ \mathtt{type} \\ x : Ob,\ y : Ob & \vdash_{Cat}\ Mor\ x\ y & :\ \mathtt{type} \\ x : Ob & \vdash_{Cat}\ id\ x & :\ Mor\ x\ x \\ x : Ob,\ y : Ob,\ z : Ob, & & \\ \quad g : Mor\ y\ z,\ f : Mor\ x\ y & \vdash_{Cat}\ g \circ f & :\ Mor\ x\ z \\ w : Ob,\ x : Ob,\ y : Ob,\ z : Ob, & & \\ \quad f : Mor\ w\ x,\ g : Mor\ x\ y,\ h : Mor\ y\ z & \vdash_{Cat}\ h \circ (g \circ f) & \equiv\ (h \circ g) \circ f \\ x : Ob,\ y : Ob,\ f : Mor\ x\ y & \vdash_{Cat}\ f \circ id\ x & \equiv\ f \\ x : Ob,\ y : Ob,\ f : Mor\ x\ y & \vdash_{Cat}\ id\ y \circ f & \equiv\ f \end{array}$$

Here we have used two common abbreviations. (i) *Mor* is declared as $Mor : (x : Ob,\ y : Ob)\mathtt{type}$, and we abbreviate the type application $Mor\ x/s,\ y/t$ as $Mor\ s\ t$. (ii) $\circ$ is declared as a constant

$$\circ\ :\ \Pi_{x:Ob}\,\Pi_{y:Ob}\,\Pi_{z:Ob}\,\Pi_{g:Mor\ y\ z}\,\Pi_{f:Mor\ x\ y}\,Mor\ x\ z$$

and we abbreviate $\circ\ x\ y\ z\ g\ f$ as $g \circ f$. This is unambiguous because the values of the first three arguments can be inferred from the types of the last two arguments.

The axioms of a category are declared using the Curry-Howard equivalence ([CF58, How80]) of MLTT and intuitionistic first-order logic without negation ([See84]). For example, to obtain right-neutrality, we declare a constant

$$neutr\ :\ \Pi_{x:Ob}\,\Pi_{y:Ob}\,\Pi_{f:Mor\ x\ y}\,Id(f \circ id\ x, f)$$

Such a constant yields the corresponding equality judgment above using Rule $e_{Id(-,-)}$ from Fig. 6.

The *rules* for signatures, contexts, and substitutions are given in Fig. 3. A *signature* is a list of declarations of type-valued constants a or term constants c. For example, $a\!:\!(\Gamma)\texttt{type}$ means that a can be applied to arguments with types given by Γ and returns a type. The domain of a signature is defined by $\mathrm{dom}(\cdot) = \varnothing$, $\mathrm{dom}(\Sigma, a\!:\!(\Gamma)\texttt{type}) = \mathrm{dom}(\Sigma) \cup \{a\}$, and $\mathrm{dom}(\Sigma, c\!:\!S) = \mathrm{dom}(\Sigma) \cup \{c\}$.

Contexts are similar to signatures except that they only declare variables ranging over terms. The domain of a context is defined as for signatures. A *substitution* from Γ to Γ' is a list of terms in context Γ' such that each term is typed by the corresponding type in Γ. Note that in a context $x_1\!:\!S_1, \ldots, x_n\!:\!S_n$, the variable x_i may occur in $S_{i+1}, \ldots, S_n$.

$$\frac{}{\vdash \cdot\ \texttt{Sig}}\,\Sigma_{\cdot} \qquad \frac{\vdash \Sigma\ \texttt{Sig} \quad \cdot \vdash_\Sigma S : \texttt{type} \quad c \notin \mathrm{dom}(\Sigma)}{\vdash \Sigma, c\!:\!S\ \texttt{Sig}}\,\Sigma_c$$

$$\frac{\vdash \Sigma\ \texttt{Sig} \quad \vdash_\Sigma \Gamma'\ \texttt{Ctx} \quad a \notin \mathrm{dom}(\Sigma)}{\vdash \Sigma, a\!:\!(\Gamma')\texttt{type}\ \texttt{Sig}}\,\Sigma_a$$

$$\frac{\vdash \Sigma\ \texttt{Sig}}{\vdash_\Sigma \cdot\ \texttt{Ctx}}\,\Gamma_{\cdot} \qquad \frac{\vdash_\Sigma \Gamma\ \texttt{Ctx} \quad \Gamma \vdash_\Sigma S : \texttt{type} \quad x \notin \mathrm{dom}(\Gamma)}{\vdash_\Sigma \Gamma, x\!:\!S\ \texttt{Ctx}}\,\Gamma_x$$

$$\frac{\vdash_\Sigma \Gamma'\ \texttt{Ctx}}{\vdash_\Sigma \cdot : \cdot \to \Gamma'}\,\sigma_{\cdot} \qquad \frac{\vdash_\Sigma \gamma : \Gamma \to \Gamma' \quad \Gamma \vdash_\Sigma S : \texttt{type} \quad \Gamma' \vdash_\Sigma s : \gamma(S)}{\vdash_\Sigma \gamma, x/s : \Gamma, x\!:\!S \to \Gamma'}\,\sigma_x$$

Figure 3: Signatures, Contexts, Substitutions

Fig. 4 gives the formation rules for *types*. In context Γ, an application $a\ \gamma_0$ of a type constructor $a\!:\!(\Gamma_0)\texttt{type}$ to a substitution γ_0 from Γ_0 into Γ, means that γ_0 provides a list of terms as arguments to a.

Fig. 5 gives the *term* formation rules. For the case where only one variable is to be substituted in an expression e in context $\Gamma, x\!:\!S$, we define

$$e[x/s] := (\mathrm{id}_\Gamma, x/s)(e).$$

We have the following subexpression property: $\Gamma \vdash_\Sigma s : S$ implies $\Gamma \vdash_\Sigma S : \texttt{type}$ implies $\vdash_\Sigma \Gamma\ \texttt{Ctx}$ implies $\vdash \Sigma\ \texttt{Sig}$.

$$\frac{a : (\Gamma_0)\mathtt{type} \text{ in } \Sigma \quad \vdash_\Sigma \gamma_0 : \Gamma_0 \to \Gamma}{\Gamma \vdash_\Sigma a\ \gamma_0 : \mathtt{type}}\, T_{app}$$

$$\frac{\vdash_\Sigma \Gamma\ \mathtt{Ctx}}{\Gamma \vdash_\Sigma 1 : \mathtt{type}}\, T_1 \qquad \frac{\Gamma \vdash_\Sigma s : S \quad \Gamma \vdash_\Sigma s' : S}{\Gamma \vdash_\Sigma Id(s, s') : \mathtt{type}}\, T_{Id(-,-)}$$

$$\frac{\Gamma, x : S \vdash_\Sigma T : \mathtt{type}}{\Gamma \vdash_\Sigma \Sigma_{x:S}\, T : \mathtt{type}}\, T_\Sigma \qquad \frac{\Gamma, x : S \vdash_\Sigma T : \mathtt{type}}{\Gamma \vdash_\Sigma \Pi_{x:S}\, T : \mathtt{type}}\, T_\Pi$$

Figure 4: Types

$$\frac{c : S \text{ in } \Sigma \quad \vdash_\Sigma \Gamma\ \mathtt{Ctx}}{\Gamma \vdash_\Sigma c : S}\, t_c \qquad \frac{\vdash_\Sigma \Gamma\ \mathtt{Ctx} \quad x : S \text{ in } \Gamma}{\Gamma \vdash_\Sigma x : S}\, t_x$$

$$\frac{\vdash_\Sigma \Gamma\ \mathtt{Ctx}}{\Gamma \vdash_\Sigma * : 1}\, t_* \qquad \frac{\Gamma \vdash_\Sigma s : S}{\Gamma \vdash_\Sigma \mathit{refl}(s) : Id(s, s)}\, t_{\mathit{refl}(-)}$$

$$\frac{\Gamma \vdash_\Sigma s : S \quad \Gamma, x : S \vdash_\Sigma T : \mathtt{type} \quad \Gamma \vdash_\Sigma t : T[x/s]}{\Gamma \vdash_\Sigma \langle s, t \rangle : \Sigma_{x:S}\, T}\, t_{\langle -,- \rangle}$$

$$\frac{\Gamma \vdash_\Sigma u : \Sigma_{x:S}\, T}{\Gamma \vdash_\Sigma \pi_1(u) : S}\, t_{\pi_1} \qquad \frac{\Gamma \vdash_\Sigma u : \Sigma_{x:S}\, T}{\Gamma \vdash_\Sigma \pi_2(u) : T[x/\pi_1(s)]}\, t_{\pi_2}$$

$$\frac{\Gamma, x : S \vdash_\Sigma t : T}{\Gamma \vdash_\Sigma \lambda_{x:S}\, t : \Pi_{x:S}\, T}\, t_\lambda \qquad \frac{\Gamma \vdash_\Sigma f : \Pi_{x:S}\, T \quad \Gamma \vdash_\Sigma s : S}{\Gamma \vdash_\Sigma f\ s : T[x/s]}\, t_{app}$$

Figure 5: Terms

Fig. 6 gives the congruence and conversion rules for the *equality of terms*. η-conversion, reflexivity, symmetry, transitivity, and congruence rules for the other term constructors are omitted because they are derivable or admissible. In particular, η-conversion is implied by functional extensionality e_{funcext}. The rules have extra premises ensuring well-formedness of subexpressions, but these are elided for ease of reading, i.e., we assume that all terms occurring in Fig. 6 are well-formed without making that explicit in the rules.

Finally, Fig. 7 gives a simple axiomatization of the equality of types. Note that equality of types is decidable iff the equality of terms is.

$$\frac{\Gamma \vdash_\Sigma v : Id(s,s')}{\Gamma \vdash_\Sigma s \equiv s'}\, e_{Id(-,-)} \qquad \frac{\Gamma \vdash_\Sigma v : Id(s,s') \quad \Gamma \vdash_\Sigma v' : Id(s,s')}{\Gamma \vdash_\Sigma v \equiv v'}\, e_{id-uniq}$$

$$\frac{\Gamma \vdash_\Sigma s : 1}{\Gamma \vdash_\Sigma s \equiv *}\, e_* \qquad \frac{}{\Gamma \vdash_\Sigma \langle \pi_1(u), \pi_2(u)\rangle \equiv u}\, e_{\langle -,-\rangle}$$

$$\frac{}{\Gamma \vdash_\Sigma \pi_1(\langle s, s'\rangle) \equiv s}\, e_{\pi_1} \qquad \frac{}{\Gamma \vdash_\Sigma \pi_1(\langle s, s'\rangle) \equiv s'}\, e_{\pi_2}$$

$$\frac{}{\Gamma \vdash_\Sigma (\lambda_{x:S}\, t)\ s \equiv t[x/s]}\, e_\beta \qquad \frac{\Gamma \vdash_\Sigma f \equiv f' \quad \Gamma \vdash_\Sigma s \equiv s'}{\Gamma \vdash_\Sigma f\ s \equiv f'\ s'}\, e_{app}$$

$$\frac{\Gamma \vdash_\Sigma f : \Pi_{x:S} T \quad \Gamma \vdash_\Sigma f' : \Pi_{x:S} T \quad \Gamma, y{:}S \vdash_\Sigma f\ y \equiv f'\ y}{\Gamma \vdash_\Sigma f \equiv f'}\, e_{funcext}$$

$$\frac{\Gamma \vdash_\Sigma s : S \quad \Gamma \vdash_\Sigma s \equiv s' \quad \Gamma \vdash_\Sigma S \equiv S'}{\Gamma \vdash_\Sigma s' : S'}\, e_{typing}$$

Figure 6: Equality of Terms

$$\frac{\begin{array}{l}\gamma = x_1/s_1, \ldots, x_n/s_n \\ \gamma' = x_1/s'_1, \ldots, x_n/s'_n\end{array} \quad \Gamma \vdash_\Sigma s_i \equiv s'_i \ \text{ for } i = 1, \ldots, n}{\Gamma \vdash_\Sigma a\ \gamma \equiv a\ \gamma'}\, E_a$$

$$\frac{}{\Gamma \vdash_\Sigma 1 \equiv 1}\, E_1 \qquad \frac{\Gamma \vdash_\Sigma s_1 \equiv s'_1 \quad \Gamma \vdash_\Sigma s_2 \equiv s'_2}{\Gamma \vdash_\Sigma Id(s_1, s_2) \equiv Id(s'_1, s'_2)}\, E_{Id(-,-)}$$

$$\frac{\Gamma \vdash_\Sigma S \equiv S' \quad \Gamma, x{:}S \vdash_\Sigma T \equiv T'}{\Gamma \vdash_\Sigma \Sigma_{x:S}\, T \equiv \Sigma_{x:S'}\, T'}\, E_\Sigma \qquad \frac{\Gamma \vdash_\Sigma S \equiv S' \quad \Gamma, x{:}S \vdash_\Sigma T \equiv T'}{\Gamma \vdash_\Sigma \Pi_{x:S}\, T \equiv \Pi_{x:S'}\, T'}\, E_\Pi$$

Figure 7: Equality of Types

Parallel to Def. 2.1, we obtain the following basic property of substitutions by a straightforward induction on derivations:

Lemma 2.3. *Assume* $\vdash_\Sigma \gamma : \Gamma \to \Gamma'$. *Then:*

$$\begin{array}{llll} \textit{if} & \vdash_\Sigma \delta : \Delta \to \Gamma & \textit{then} & \vdash_\Sigma \gamma \circ \delta : \Delta \to \Gamma', \\ \textit{if} & \Gamma \vdash_\Sigma S : \mathtt{type} & \textit{then} & \Gamma' \vdash_\Sigma \gamma(S) : \mathtt{type}, \\ \textit{if} & \Gamma \vdash_\Sigma s : S & \textit{then} & \Gamma' \vdash_\Sigma \gamma(s) : \gamma(S). \end{array}$$

3. Categorical Preliminaries

In this section, we repeat some well-known definitions and results about indexed sets and fibrations over posets (see, e.g., [Joh02]). We assume the basic notions of category theory (see, e.g., [Mac98]). We use a set-theoretical pairing function (a, b) and define tuples as left-associatively nested pairs, i.e., $(a_1, a_2, \ldots, a_n)$ abbreviates $(\ldots(a_1, a_2), \ldots, a_n)$.

Definition 3.1 (Indexed Sets)**.** $\mathcal{POSET}$ denotes the category of partially ordered sets. We treat posets as categories and write $p \le p'$ for the uniquely determined morphism $p \to p'$. If P is a poset, $\mathcal{SET}^P$ denotes the category of functors $P \to \mathcal{SET}$ and natural transformations. These functors are also called *P-indexed sets*.

We denote the constant P-indexed set that maps each $p \in P$ to $\{\varnothing\}$ by 1_P. It is often convenient to replace an indexed set A over P with a poset formed from the disjoint union of all sets $A(p)$ for $p \in P$. This is a special case of the category of elements, a construction due to Mac Lane ([MM92]) that is sometimes also called the Grothendieck construction.

Definition 3.2 (Category of Elements)**.** For an indexed set A over P, we define a poset $\int_P A := \{(p, a) \mid p \in P, a \in A(p)\}$ with

$$(p, a) \le (p', a') \quad \text{iff} \quad p \le p' \text{ and } A(p \le p')(a) = a'.$$

We also write $\int A$ instead of $\int_P A$ if P is clear from the context.

Using the category of elements, we can work with sets indexed by indexed sets: We write $P|A$ if A is an indexed set over P, and $P|A|B$ if additionally B is an indexed set over $\int_P A$, etc.

Definition 3.3. Assume $P|A|B$. We define an indexed set $P|(A \ltimes B)$ by

$$(A \ltimes B)(p) = \{(a, b) \mid a \in A(p), b \in B(p, a)\}$$

and

$$(A \ltimes B)(p \le p') : (a, b) \mapsto \big(a', B\big((p, a) \le (p', a')\big)(b)\big) \qquad \text{for } a' = A(p \le p')(a).$$

And we define a natural transformation $\pi_B : A \ltimes B \to A$ by

$$(\pi_B)_p : (a, b) \mapsto a.$$

The following definition introduces discrete opfibrations; for brevity, we will refer to them as "fibrations" in the sequel. Using the axiom of choice, these are necessarily split.

Definition 3.4 (Fibrations)**.** A *fibration* over a poset P is a functor $f : Q \to P$ for a poset Q with the following property: For all $p' \in P$ and $q \in Q$ such that $f(q) \le p'$, there is a unique $q' \in Q$ such that $q \le q'$ and $f(q') = p'$. We call f *canonical* iff f is the first projection of $Q = \int_P A$ for some $P|A$.

For every indexed set A over P, the first projection $\int_P A \to P$ is a (canonical) fibration. Conversely, every fibration $f : Q \to P$ defines an indexed set over P by mapping $p \in P$ to its preimage $f^{-1}(p) \subseteq Q$ and $p \le p'$ to the obvious function. This leads to a well-known equivalence of indexed sets and fibrations over P. If we only consider canonical fibrations, we obtain an isomorphism as follows.

Lemma 3.5. *If we restrict the objects of $\mathcal{POSET}/P$ to be canonical fibrations and the morphisms to be (arbitrary) fibrations, we obtain the full subcategory* $\mathrm{Fib}(P)$ *of $\mathcal{POSET}/P$. There are isomorphisms*

$$F(-) : \mathcal{SET}^P \to \mathrm{Fib}(P) \quad \text{and} \quad I(-) : \mathrm{Fib}(P) \to \mathcal{SET}^P.$$

Proof. It is straightforward to show that $\mathrm{Fib}(P)$ is a full subcategory: The identity in $\mathcal{POSET}$ and the composition of two fibrations are fibrations. Thus, it only remains to show that if $f \circ \varphi = f'$ in $\mathcal{POSET}$ where f and f' are fibrations and φ is a morphism in $\mathcal{POSET}$, then φ is a fibration as well. This is easy.

For $A : P \to \mathcal{SET}$, we define the fibration $F(A) : \int_P A \to P$ by $(p, a) \mapsto p$. And for a natural transformation $\eta : A \to A'$, we define the fibration $F(\eta) : \int_P A \to \int_P A'$ satisfying $F(A) \circ F(\eta) = F(A')$ by $(p, a) \mapsto (p, \eta_p(a))$.

For $f : Q \to P$, we obtain an indexed set using the fact that f is canonical. More concretely, we define $I(f)(p) := \{a \mid f(p, a) = p\}$ and $I(f)(p \leq p') : a \mapsto a'$ where a' is the uniquely determined element such that $(p, a) \leq (p', a') \in Q$. And for a morphism φ between fibrations $f : Q \to P$ and $f' : Q' \to P$, we define a natural transformation $I(\varphi) : I(f) \to I(f')$ by $I(\varphi)_p : a \mapsto a'$ where a' is such that $\varphi(p, a) = (p, a')$.

Then it is easy to compute that I and F are mutually inverse functors. □

Definition 3.6 (Indexed Elements)**.** Assume $P|A$. The *P-indexed elements* of A are given by

$$\mathrm{Elem}(A) := \left\{ \left(a_p \in A(p)\right)_{p \in P} \mid a_{p'} = A(p \leq p')(a_p) \text{ whenever } p \leq p' \right\}.$$

Then the indexed elements of A are in bijection with the natural transformations $1_P \to A$. For $a \in \mathrm{Elem}(A)$, we will write $F(a)$ for the fibration $P \to \int A$ mapping p to (p, a_p). $F(a)$ is a section of $F(A)$, and indexed elements are also called global sections.

Example 3.7. We exemplify the introduced notions by Fig. 8. P is a totally ordered set visualized as a horizontal line with two elements $p_1 \leq p_2 \in P$. For $P|A$, $\int A$ becomes a blob over P. The sets $A(p_i)$ correspond to the vertical lines in $\int A$, and $a_i \in A(p_i)$. The action of $A(p \leq p')$ and the poset structure of $\int A$ are horizontal: If we assume $A(p_1 \leq p_2) : a_1 \mapsto a_2$, then $(p_1, a_1) \leq (p_2, a_2)$ in $\int A$. Finally, the action of $F(A)$ is vertical: $F(A)$ maps (p_i, a_i) to p_i. Note that our intuitive visualization is not meant to indicate that the sets $A(p_i)$ must be in bijection or that the mapping $A(p_1 \leq p_2)$ must be injective or surjective.

Similarly, for $P|A|B$, $\int B$ becomes a three-dimensional blob over $\int A$. The sets $B(p_i, a_i)$ correspond to the dotted lines. Again the action of $B((p_1, a_1) \leq (p_2, a_2))$ and the poset structure of $\int B$ are horizontal:

$$b_i \in B(p_i, a_i) \quad \text{and} \quad B((p_1, a_1) \leq (p_2, a_2)) : b_1 \mapsto b_2$$

and $F(B)$ projects vertically from $\int B$ to $\int A$.

Similarly, we have

$$(a_i, b_i) \in (A \ltimes B)(p_i) \quad \text{and} \quad (A \ltimes B)(p_1 \leq p_2) : (a_1, b_1) \mapsto (a_2, b_2)$$

Thus, the sets $(A \ltimes B)(p_i)$ correspond to the two-dimensional gray areas. The sets $\int_P (A \ltimes B)$ and $\int_{\int_P A} B$ are isomorphic, and their elements differ only in the bracketing:

$$(p_i, (a_i, b_i)) \in \textstyle\int_P (A \ltimes B) \quad \text{and} \quad ((p_i, a_i), b_i) \in \int_{\int_P A} B.$$

Up to this isomorphism, the projection $F(A \ltimes B)$ is the composite $F(A) \circ F(B)$.

Indexed elements $a \in \mathrm{Elem}(A)$ are families $(a_p)_{p\in P}$ and correspond to horizontal curves through $\int A$ such that $F(a)$ is a section of $F(A)$. Indexed elements of B correspond to two-dimensional vertical areas in $\int B$ (intersecting each line parallel to the dotted lines exactly once), and indexed elements of $A \ltimes B$ correspond to horizontal curves in $\int B$ (intersecting each area parallel to the gray areas exactly once).

Finally the condition that indexed elements are natural transformations can be visualized as follows: The indexed elements $a \in \mathrm{Elem}(A)$ are exactly those horizontal curves that arise if a line is drawn from (p, a) to (p', a') whenever $(p, a) \leq (p', a')$. There may be multiple such curves going through a point (p, a), but they must coincide to the right of (p, a). Moreover, $(p, a) \leq (p', a')$ holds iff (p, a) is to the left of (p', a') on the same curve. In particular, if P has a least element p_0, we obtain exactly one such curve for every element of $A(p_0)$.

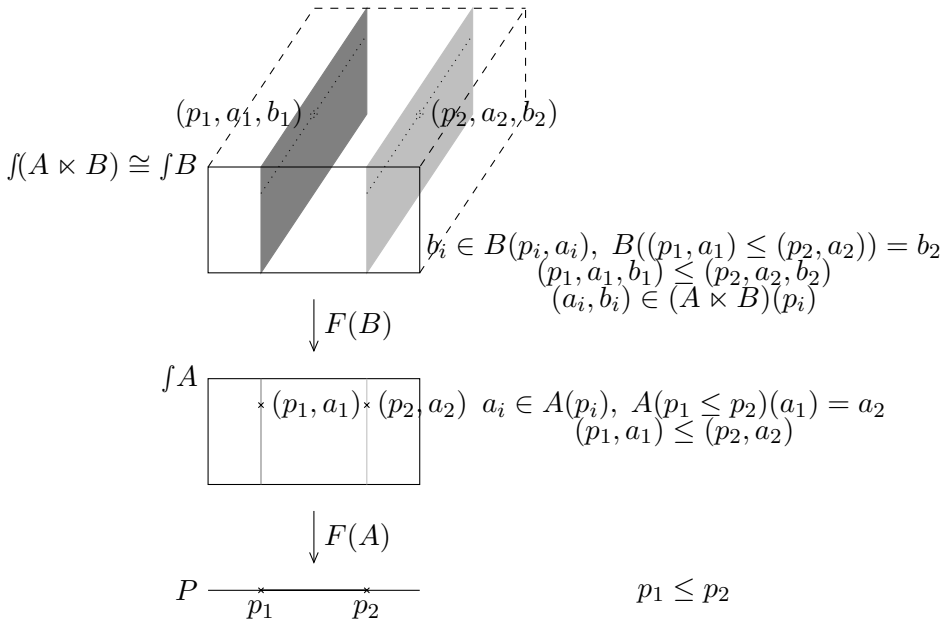


Figure 8: Indexed Sets and Fibrations

Example 3.8. Let *Sign* be the set of well-formed signatures of MLTT (or of any other type theory for that matter). *Sign* is a poset under inclusion $\subseteq$ of signatures. Let $Con(\Sigma)$ be the set of well-formed contexts over Σ, and let $Con(\Sigma \subseteq \Sigma') : Con(\Sigma) \hookrightarrow Con(\Sigma')$ be an inclusion. Then $Sign|Con$, and the tuple assigning the empty context to every signature is an example of an indexed element of *Con*.

$\int_{Sign} Con$ is the set of pairs (Σ, Γ) such that $\vdash_\Sigma \Gamma$ `Ctx`, and $(\Sigma, \Gamma) \leq (\Sigma', \Gamma')$ iff $\Sigma \subseteq \Sigma'$ and $\Gamma = \Gamma'$. Let $Typ(\Sigma, \Gamma)$ be the set of types S such that $\Gamma \vdash_\Sigma S$: `type`. *Typ* becomes an indexed set $Sign|Con|Typ$ by defining $Typ((\Sigma, \Gamma) \leq (\Sigma', \Gamma))$ to be an inclusion. The tuple assigning 1 to every pair (Σ, Γ) is an example of an indexed element of *Typ*.

We will use Lem. 3.5 frequently to switch between indexed sets and fibrations, as convenient. In particular, we will use the following two corollaries.

Lemma 3.9. *Assume $P|A$. Then*

$$\mathrm{Elem}(A) \cong \mathrm{Hom}_{\mathrm{Fib}(P)}(\mathrm{id}_P, F(A)) = \{f : P \to \textstyle\int_P A \mid F(A) \circ f = \mathrm{id}_P\}.$$

and

$$\mathcal{SET}^P/A \cong \mathcal{SET}^{\int A}$$

Proof. Both claims follow from Lem. 3.5 by using $\mathrm{Elem}(A) \cong \mathrm{Hom}_{\mathcal{SET}^P}(1_P, A)$ as well as $\mathrm{Fib}(P)/F(A) \cong \mathrm{Fib}(\int_P A)$, respectively. □

Finally, as usual, we say that a category is *locally cartesian closed* (LCC) if it and all of its slice categories are cartesian closed (in particular, it has a terminal object). Then we have the following well-known result.

Lemma 3.10. *$\mathcal{SET}^P$ is LCC.*

Proof. The terminal object is given by 1_P. The product is taken pointwise: $A \times B : p \mapsto A(p) \times B(p)$ and similarly for morphisms. The exponential object is given by: $B^A : p \mapsto \mathrm{Hom}_{\mathcal{SET}^{P^p}}(A^p, B^p)$ where A^p and B^p are as A and B but restricted to $P^p := \{p' \in P \mid p \leq p'\}$. $B^A(p \leq p')$ maps a natural transformation, which is a family of mappings over P^p, to its restriction to $P^{p'}$. This proves that $\mathcal{SET}^P$ and so also $\mathrm{Fib}(P)$ is cartesian closed for any P. By Lem. 3.9, we obtain the same for all slice categories. □

4. Operations on Indexed Sets

Because $\mathcal{SET}^P$ is LCC, we know that it has pullbacks and that the pullback along a fixed natural transformation has left and right adjoints (see, e.g., [Joh02]). However, these functors are only unique up to isomorphism, and it is non-trivial to pick coherent choices for them.

Pullbacks. Assume $P|A_1$ and $P|A_2$ and a natural transformation $h : A_2 \to A_1$. The pullback along h is a functor $\mathcal{SET}^P/A_1 \to \mathcal{SET}^P/A_2$. Using Lem. 3.9, we can avoid dealing with slice categories of $\mathcal{SET}^P$ and instead give a functor

$$h^* : \mathcal{SET}^{\int A_1} \to \mathcal{SET}^{\int A_2},$$

which we also call the pullback along h. The functor h^* is given by precomposition:

Definition 4.1. Assume A_1 and A_2 indexed over P, and a natural transformation $h : A_2 \to A_1$. Then for $B \in \mathcal{SET}^{\int A_1}$, we put

$$h^*B := B \circ F(h) \in \mathcal{SET}^{\int A_2},$$

where, as in Lem. 3.5, $F(h) : \int_P A_2 \to \int_P A_1$. The action of h^* on morphisms is defined similarly by composing a natural transformation $\beta : B \to B'$ with the functor $F(h)$: $h^*\beta := \beta \circ F(h)$. Finally, we define a natural transformation between P-indexed sets by

$$h \ltimes B : A_2 \ltimes h^*B \to A_1 \ltimes B, \quad (h \ltimes B)_p : (a_2, b) \mapsto (h_p(a_2), b).$$

The application of $h \ltimes B$ is independent of B, which is only needed in the notation to determine the domain and codomain of $h \ltimes B$.

Lemma 4.2 (Pullbacks)**.** *In the situation of Def. 4.1, the following is a pullback in* $\mathcal{SET}^P$*.*

$$\begin{array}{ccc} A_2 \ltimes h^*B & \xrightarrow{h \ltimes B} & A_1 \ltimes B \\ \downarrow{\scriptstyle \pi_{h^*B}} & & \downarrow{\scriptstyle \pi_B} \\ A_2 & \xrightarrow{\quad h \quad} & A_1 \end{array}$$

Furthermore, we have the following coherence properties for every natural transformation $g : A_3 \to A_2$*:*

$$(\mathrm{id}_{A_1})^*B = B, \quad \mathrm{id}_{A_1} \ltimes B = \mathrm{id}_{A_1 \ltimes B},$$
$$(h \circ g)^*B = g^*(h^*B), \quad (h \circ g) \ltimes B = (h \ltimes B) \circ (g \ltimes h^*B).$$

Proof. The following is a pullback in $\mathcal{POSET}$:

$$\begin{array}{ccc} \int A_2 \ltimes h^*B & \xrightarrow{F(h \ltimes B)} & \int A_1 \ltimes B \\ \downarrow{\scriptstyle F(\pi_{h^*B})\ \ F(\pi_B)} & & \downarrow \\ \int A_2 & \xrightarrow{\quad F(h) \quad} & \int A_1 \end{array} \qquad \begin{array}{ccc} (p,(a_2,b)) & \xmapsto{F(h \ltimes B)} & (p,(h_p(a_2),b)) \\ \downarrow{\scriptstyle F(\pi_{h^*B})} \quad {\scriptstyle F(\pi_B)}\downarrow & & \\ (p,a_2) & \xmapsto{\quad F(h) \quad} & (p,h_p(a_2)) \end{array}$$

If we turn this square into a cocone on P by adding the canonical projections $F(A_2)$ and $F(A_1)$, it becomes a pullback in $\mathrm{Fib}(P)$. Then the result follows by Lem. 3.5. The coherence properties can be verified by simple computations. □

Equivalently, using the terminology of [Pit00], we can say that for every P the tuple

$$(\mathcal{SET}^P, \mathcal{SET}^{\int A}, A \ltimes B, \pi_B, h^*B, h \ltimes B)$$

forms a type category (where A, B, h indicate arbitrary arguments). Then giving coherent adjoints to the pullback functor shows that this type category admits dependent sums and products.

Adjoints. To interpret MLTT, the adjoints to h^*, where $h : A_2 \to A_1$, are only needed if h is a projection, i.e., $A_1 := A$, $A_2 := A \ltimes B$, and $h := \pi_B$ for some $P|A|B$. We only give adjoint functors for this special case because we use this restriction when defining the right adjoint. Thus, we give functors

$$\mathcal{L}_B, \mathcal{R}_B : \mathcal{SET}^{\int A \ltimes B} \to \mathcal{SET}^{\int A} \text{ such that } \mathcal{L}_B \dashv {\pi_B}^* \dashv \mathcal{R}_B$$

in Def. 4.3 and 4.6, respectively. These functors will satisfy the coherence properties

$$g^*(\mathcal{L}_B C) = \mathcal{L}_{g^*B}(g \ltimes B)^*C \quad \text{and} \quad g^*(\mathcal{R}_B C) = \mathcal{R}_{g^*B}(g \ltimes B)^*C$$

for every $g : A' \to A$, which we prove in Lem. 4.4 and 4.7, respectively.

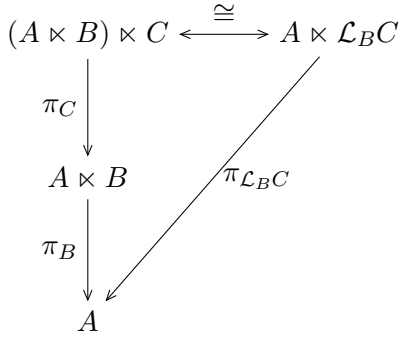
Definition 4.3. We define the functor $\mathcal{L}_B$ as follows. For an object C, we put $\mathcal{L}_B C := B \ltimes (C \circ \mathit{assoc})$ where *assoc* maps elements $((p,a),b) \in \int B$ to $(p,(a,b)) \in \int A \ltimes B$; and for a morphism, i.e., a natural transformation $\eta : C \to C'$, we put

$$(\mathcal{L}_B \eta)_{(p,a)} : (b,c) \mapsto (b, \eta_{(p,(a,b))}(c)) \qquad \text{for } (p,a) \in \int A.$$

Lemma 4.4 (Left Adjoint)**.** *$\mathcal{L}_B$ is left adjoint to $\pi_B{}^*$. Furthermore, for any natural transformation $g : A' \to A$, we have the following coherence property (the Beck-Chevalley condition)*

$$g^*(\mathcal{L}_B C) = \mathcal{L}_{g^*B}(g \ltimes B)^* C.$$

Proof. It is easy to show that $\mathcal{L}_B$ is isomorphic to composition along π_B, for which the adjointness is well-known. In particular, we have the following diagram in $\mathcal{SET}^P$:



The coherence can be verified by direct computation. □

The right adjoint is more complicated. Intuitively, $\mathcal{R}_B C$ must represent the dependent functions from B to C. The naive candidate for this is $\mathrm{Elem}(C) \cong \mathrm{Hom}(1_{\int B}, C)$ (i.e., $\mathrm{Hom}(B, C)$ in the simply-typed case), but this is not a $\int A$-indexed set. There is a well-known construction to remedy this, but we use a subtle modification to achieve coherence, i.e., the corresponding Beck-Chevalley condition. To do that, we need an auxiliary definition.

Definition 4.5. Assume $P|A|B$, $P|(A \ltimes B)|C$, and an element $x := (p, a) \in \int A$. Let $A^x \in \mathcal{SET}^P$ and a natural transformation $\mathrm{i}^x : A^x \to A$ be given by

$$A^x(p') = \begin{cases} \{\varnothing\} & \text{if } p \le p' \\ \varnothing & \text{otherwise} \end{cases} \qquad \mathrm{i}^x_{p'} : \varnothing \mapsto A(p \le p')(a).$$

Then we define indexed sets $P|A^x|B^x$ and $P|(A^x \ltimes B^x)|C^x$ by:

$$B^x := \mathrm{i}^{x*} B, \quad C^x := (\mathrm{i}^x \ltimes B)^* C$$

and put $d^x := \int A^x \ltimes B^x$ for the domain of C^x.

Note that A^x is the Yoneda embedding of p in $\mathcal{SET}^P$. The left diagram in Fig. 9 shows the involved P-indexed sets, the right one gives the actions of the natural transformations for an element $p' \in P$ with $p \le p'$. Below it will be crucial for coherence that B^x and C^x contain tuples in which a' is replaced with $\varnothing$.

Definition 4.6. Assume $P|A|B$. Then we define the functor $\mathcal{R}_B : \mathcal{SET}^{\int A \ltimes B} \to \mathcal{SET}^{\int A}$ as follows. Firstly, for an object C, we put for $x \in \int A$

$$(\mathcal{R}_B C)(x) := \mathrm{Elem}(C^x).$$

In particular, $f \in (\mathcal{R}_B C)(x)$ is a family $(f_y)_{y \in d^x}$ with $f_y \in C^x(y)$. For $x \le x' \in \int A$, we have $d^x \supseteq d^{x'}$ and put

$$(\mathcal{R}_B C)(x \le x') : (f_y)_{y \in d^x} \mapsto (f_y)_{y \in d^{x'}}.$$

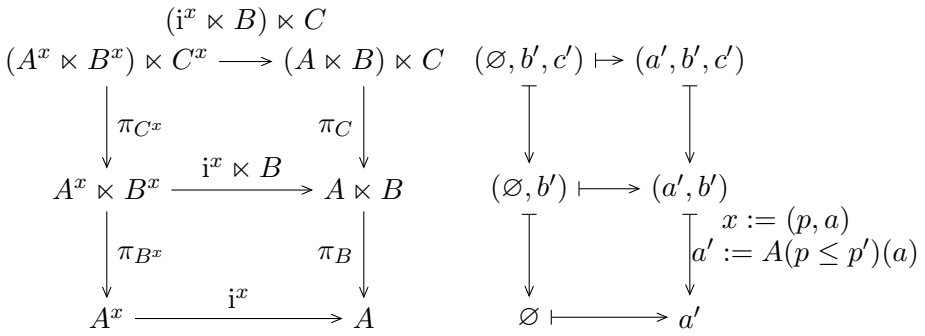


Figure 9: The Situation of Def. 4.5

Secondly, for a morphism, i.e., a natural transformation $\eta : C \to C'$, we define $\mathcal{R}_B\eta : \mathcal{R}_B C \to \mathcal{R}_B C'$ as follows: For $x := (p, a) \in \int A$ and $f \in (\mathcal{R}_B C)(x)$, we define $f' := (\mathcal{R}_B\eta)_x(f) \in (\mathcal{R}_B C')(x)$ by

$$f'_{(p',(\varnothing,b'))} := \eta_{(p',(a',b'))}(f_{(p',(\varnothing,b'))}) \qquad \text{for } (p', (\varnothing, b')) \in d^x \text{ and } a' := A(p \leq p')(a).$$

Lemma 4.7 (Right Adjoint)**.** *$\mathcal{R}_B$ is right adjoint to ${\pi_B}^*$. Furthermore, for every natural transformation $g : A' \to A$, we have the following coherence property*

$$g^*(\mathcal{R}_B C) = \mathcal{R}_{g^*B}(g \ltimes B)^* C.$$

Proof. Assume $P|A|B$, $P|A \ltimes B|C$, and $x = (p, a) \in \int A$. Let $\mathrm{y}(x) \in \mathcal{SET}^{\int A}$ be the covariant representable functor of x mapping $x' \in \int A$ to a singleton iff $x \leq x'$ and to the empty set otherwise. Since we know the right adjoint exists, we can use the Yoneda lemma for covariant functors to derive sufficient and necessary constraints for $\mathcal{R}_B$ to be a right adjoint:

$$\begin{aligned}(\mathcal{R}_B C)(x) &\cong \mathrm{Hom}_{\mathcal{SET}^{\int A}}(\mathrm{y}(x), \mathcal{R}_B C) \cong \mathrm{Hom}_{\mathcal{SET}^{\int A \ltimes B}}({\pi_B}^*\mathrm{y}(x), C)\\ &\cong \mathrm{Hom}_{\mathrm{Fib}(\int A \ltimes B)}(F({\pi_B}^*\mathrm{y}(x)), F(C)).\end{aligned}$$

Let i^x be as in Def. 4.5. Let $\mathrm{Fib}'(Q)$ be the category of (not necessarily canonical) fibrations on Q. Then it is easy to check that $F(\mathrm{i}^x \ltimes B)$ seen as a fibration with domain d^x and $F({\pi_B}^*\mathrm{y}(x))$ are isomorphic in $\mathrm{Fib}'(\int A \ltimes B)$. (They are not isomorphic in $\mathrm{Fib}(\int B)$ because the former is not canonical and thus not an object of $\mathrm{Fib}(\int B)$.) Using the fullness of $\mathrm{Fib}(Q)$, we obtain

$$\begin{aligned}(\mathcal{R}_B C)(x) &\cong \mathrm{Hom}_{\mathrm{Fib}'(\int A \ltimes B)}(F(\mathrm{i}^x \ltimes B), F(C))\\ &= \{f : d^x \to \int C \mid F(C) \circ f = F(\mathrm{i}^x \ltimes B)\}.\end{aligned}$$

And using the definition of C^x as a pullback, we obtain

$$(\mathcal{R}_B C)(x) \cong \{f : d^x \to \int C^x \mid F(C^x) \circ f = \mathrm{id}_{d^x}\} \cong \mathrm{Elem}(C^x).$$

And this is indeed how $\mathcal{R}_B C$ is defined. The value of $\mathcal{R}_B C$ on morphisms is verified similarly.

To show the coherence property, we assume $P|A'$, $g : A' \to A$, and $x' := (p, a') \in \int A'$. We abbreviate as follows: $a := g_p(a')$, $x := (p, a)$, $B' := g^*B$, and $C' := (g \ltimes B)^* C$. Furthermore, we write $\mathrm{i}^{x'}$, $A'^{x'}$, $B'^{x'}$, and $C'^{x'}$ according to Def. 4.5. Note that $A'^{x'} = A^x$.

Now coherence requires $g^*\mathcal{R}_B C = \mathcal{R}_{B'} C'$. And that follows if we show that

$$B'^{x'} = B^x \qquad \text{and} \qquad C'^{x'} = C^x.$$

Using Lem. 4.2, this follows from $g \circ \mathrm{i}^{x'} = \mathrm{i}^{x}$, which is an equality between natural transformations from $A^x = A'^{x'}$ to A in $\mathcal{SET}^P$. And to verify the latter, assume $o \in P$. The maps $g_o \circ \mathrm{i}^{x'}_o$ and i^x_o have domain $\varnothing$ or $\{\varnothing\}$. In the former case, there is nothing to prove. In the latter case, put

$$a'_o := \mathrm{i}^{x'}_o(\varnothing) = A'(p \le o)(a') \qquad \text{and} \qquad a_o := \mathrm{i}^x_o(\varnothing) = A(p \le o)(a).$$

Then we need to show $g_o(a'_o) = a_o$. And that is indeed the case because of the naturality of g as indicated in

$$\begin{array}{ccc} a' & \xmapsto{A'(p \le o)} & a'_o \\ \Big\downarrow{\scriptstyle g_p} & & \Big\downarrow{\scriptstyle g_o} \\ a & \xmapsto{A(p \le o)} & a_o \end{array}$$

□

Example 4.8 (Continuing Ex. 3.8)**.** The *Sign*-indexed set $Con \ltimes Typ$ maps every MLTT-signature Σ to the set of pairs (Γ, S) such that $\Gamma \vdash_\Sigma S : \mathtt{type}$. The projection π_{Typ} is a natural transformation $Con \ltimes Typ \to Con$ such that $(\pi_{Typ})_\Sigma : (\Gamma, S) \mapsto \Gamma$.

We define Tm such that $Sign|(Con \ltimes Typ)|Tm$: The set $Tm(\Sigma, (\Gamma, S))$ contains the terms s such that $\Gamma \vdash_\Sigma s : S$. $Tm((\Sigma, (\Gamma, S)) \le (\Sigma', (\Gamma, S)))$ is an inclusion.

Then we have $Sign|Con|\mathcal{L}_{Typ}\,Tm$, and $\mathcal{L}_{Typ}\,Tm$ maps (Σ, Γ) to the set of pairs (S, s) such that $\Gamma \vdash_\Sigma s : S$.

To exemplify Def. 4.5, fix an element $x = (\Sigma, \Gamma) \in \textstyle\int_{Sign} Con$. Then we have $\mathrm{i}^x_{\Sigma'}(\varnothing) = \Gamma$ for every $\Sigma \subseteq \Sigma'$. Typ^x maps the pair $(\Sigma', \varnothing)$ where $\Sigma \subseteq \Sigma'$ to $Typ(\Sigma', \mathrm{i}^x_{\Sigma'}(\varnothing)) = Typ(\Sigma', \Gamma)$. If $S \in Typ(\Sigma', \mathrm{i}^x_{\Sigma'}(\varnothing))$, then Tm^x maps $(\Sigma', (\varnothing, S))$ to the set $Tm(\Sigma', (\mathrm{i}^x_{\Sigma'}(\varnothing), S))$.

Now we have $Sign|Con|\mathcal{R}_{Typ}\,Tm$, and $\mathcal{R}_{Typ}\,Tm$ maps (Σ, Γ) to the set of indexed elements of Tm^x. Those are the families that assign to every $(\Sigma', (\varnothing, S))$ a term $s_{(\Sigma', (\varnothing, S))} \in Tm^x(\Sigma', (\varnothing, S)) = Tm(\Sigma', (\Gamma, S))$ such that $s_{(\Sigma', (\varnothing, S))} = s_{(\Sigma'', (\varnothing, S))}$ whenever $\Sigma' \subseteq \Sigma''$.

Above, we called $\mathrm{Elem}(C)$ the naive candidate for the right adjoint, and indeed the adjointness implies $\mathrm{Elem}(\mathcal{R}_B C) \cong \mathrm{Elem}(C)$. We define the isomorphisms explicitly because we will use them later on:

Lemma 4.9. *Assume $P|A|B$ and $P|(A \ltimes B)|C$. For $t \in \mathrm{Elem}(C)$ and $x := (p, a) \in \int A$, let $t^x \in \mathrm{Elem}(C^x)$ be given by*

$$(t^x)_{(p', (\varnothing, b'))} = t_{(p', (a', b'))} \qquad \text{where } a' := A(p \le p')(a).$$

And for $f \in \mathrm{Elem}(\mathcal{R}_B C)$ and $x := (p, (a, b)) \in \int A \ltimes B$, we have $f_{(p,a)} \in \mathrm{Elem}(C^x)$; thus, we can put

$$f^x := (f_{(p,a)})_{(p, (\varnothing, b))} \ \in \ C(p, (a, b)).$$

Then the sets $\mathrm{Elem}(C)$ and $\mathrm{Elem}(\mathcal{R}_B C)$ are in bijection via

$$\mathrm{Elem}(C) \ \ni \ t \ \overset{\mathrm{sp}(-)}{\longmapsto} \ (t^x)_{x \in \int A} \ \in \ \mathrm{Elem}(\mathcal{R}_B C)$$

and

$$\mathrm{Elem}(\mathcal{R}_B C) \ \ni \ f \ \overset{\mathrm{am}(-)}{\longmapsto} \ (f^x)_{x \in \int A \ltimes B} \ \in \ \mathrm{Elem}(C).$$

Proof. This follows from the right adjointness by easy computations. □

Intuitively, $\mathrm{sp}(t)$ turns $t \in \mathrm{Elem}(C)$ into a $\int A$-indexed set by splitting it into components. And $\mathrm{am}(f)$ amalgamates such a tuple of components back together. Syntactically, these operations correspond to currying and uncurrying, respectively.

Then we need one last notation. For $P|A$, indexed elements $a \in \mathrm{Elem}(A)$ behave like mappings with domain P. We can precompose such indexed elements with fibrations $f : Q \to P$ to obtain Q-indexed elements of $\mathrm{Elem}(A \circ f)$.

Definition 4.10. Assume $P|A$, $f : Q \to P$, and $a \in \mathrm{Elem}(A)$. $a * f \in \mathrm{Elem}(A \circ f)$ is defined by: $(a * f)_q := a_{f(q)}$ for $q \in Q$.

5. Semantics

Using the LCC structure developed in Sect. 4, the definition of the semantics is straightforward and well-known. To demonstrate its simplicity, we spell it out in an elementary way. The semantics is defined by induction on the derivations of the judgments listed in Fig. 2.

Firstly, for every signature $\vdash \Sigma\ \mathtt{Sig}$, we define models I, which provide interpretations $[\![c]\!]^I$ and $[\![a]\!]^I$ for all symbols declared in Σ. The models are Kripke-models, i.e., a Σ-model I is based on a poset P^I of worlds.

Secondly, I extends to an interpretation function $[\![-]\!]^I$, which interprets all Σ-expressions. We will omit the index I if no confusion is possible. $[\![-]\!]$ is such that

- if $\vdash_\Sigma \Gamma\ \mathtt{Ctx}$, then $[\![\Gamma]\!]$ is a poset (which has a canonical projection to P),
- if $\vdash_\Sigma \gamma : \Gamma \to \Gamma'$, then $[\![\gamma]\!] : [\![\Gamma']\!] \to [\![\Gamma]\!]$ is a monotone function,
- if $\Gamma \vdash_\Sigma S : \mathtt{type}$, then $[\![\Gamma|S]\!]$ is an indexed set on $[\![\Gamma]\!]$,
- if $\Gamma \vdash_\Sigma s : S$, then $[\![\Gamma|s]\!]$ is an indexed element of $[\![\Gamma|S]\!]$.

Thirdly, the judgments $\Gamma \vdash_\Sigma S \equiv S'$ and $\Gamma \vdash_\Sigma s \equiv s'$ correspond to a soundness result, which we will prove in Sect. 7.

The poset P of worlds plays the same role as the various posets $[\![\Gamma]\!]$ — it interprets the empty context. In this way, P can be regarded as interpreting an implicit or relative context. This is in keeping with the practice of type theory (and category theory), according to which closed expressions may be considered relative to some fixed but unspecified context (respectively, base category).

For a typed term $\Gamma \vdash_\Sigma s : S$, both $[\![\Gamma|s]\!]$ and $[\![\Gamma|S]\!]$ are indexed over $[\![\Gamma]\!]$. If $\Gamma = x_1 : S_1, \ldots, x_n : S_n$, an element of $[\![\Gamma]\!]$ has the form $(p, (a_1, \ldots, a_n))$ where $p \in P$ and $a_i \in [\![x_1 : S_1, \ldots, x_{i-1} : S_{i-1}|S_i]\!](p, (a_1, \ldots, a_{i-1}))$. Intuitively, a_i is an assignment to the variable x_i in world p. And if an assignment (p, α) is given, the interpretations of s and S satisfy $[\![\Gamma|s]\!]_{(p,\alpha)} \in [\![\Gamma|S]\!](p, \alpha)$. This is illustrated in the left diagram in Fig. 10.

If γ is a substitution $\Gamma \to \Gamma'$, then $[\![\gamma]\!]$ maps assignments $(p, \alpha') \in [\![\Gamma']\!]$ to assignments $(p, \alpha) \in [\![\Gamma]\!]$. And a substitution in types and terms is interpreted by pullback, i.e., composition. This is illustrated in the right diagram in Fig. 10, whose commutativity expresses the coherence. We will state this more precisely in Sect. 6.

Sum types are interpreted naturally as the dependent sum of indexed sets given by the left adjoint. And pairing and projections have their natural semantics. Product types are interpreted as exponentials using the right adjoint. A λ-abstraction $\lambda_{x:S}\, t$ is interpreted by first interpreting t and then splitting it as in Lem. 4.9. And an application $f\ s$ is interpreted by amalgamating the interpretation of f as in Lem. 4.9 and using the composition from Def. 4.10.

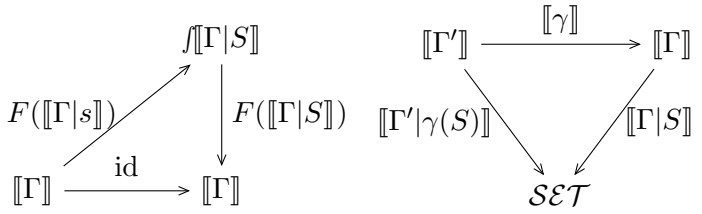


Figure 10: Semantics of Terms, Types, and Substitution

Definition 5.1 (Models)**.** For a signature Σ, Σ-*models* are defined as follows:

- A model I for the empty signature $\cdot$ is a poset P^I.
- A model I for the signature $\Sigma, c : S$ consists of a Σ-model I_Σ and an indexed element $[\![c]\!]^I \in \mathrm{Elem}([\![\cdot|S]\!]^{I_\Sigma})$.
- A model I for the signature $\Sigma, a : (\Gamma_0)\texttt{type}$ consists of a Σ-model I_Σ and an indexed set $[\![a]\!]^I$ over $[\![\Gamma_0]\!]^{I_\Sigma}$.

Definition 5.2 (Model Extension)**.** The extension of a model is defined by induction on the typing derivations. Therefore, we can assume in each case that all occurring expressions are well-formed. For example in the case for $[\![\Gamma|f\ s]\!]$, f has type $\Pi_{x:S}\,T$ and s has type S.

- Contexts: The elements of the poset $[\![x_1 : S_1, \ldots, x_n : S_n]\!]$ are the tuples $(p, (a_1, \ldots, a_n))$ such that
$$\begin{aligned} &p \in P \\ &a_1 \in [\![\cdot|S_1]\!](p, \varnothing) \\ &\vdots \\ &a_n \in [\![x_1 : S_1, \ldots, x_{n-1} : S_{n-1}|S_n]\!](p, (a_1, \ldots, a_{n-1})) \end{aligned}$$
In particular $[\![\cdot]\!] = P \times \{\varnothing\}$. The ordering of this poset is inherited from the n-times iterated category of elements, to which it is canonically isomorphic. The first projection from $[\![\Gamma]\!]$ is a canonical fibration, and we write $I([\![\Gamma]\!])$ for the corresponding indexed set.
- Substitutions $\gamma = x_1/s_1, \ldots, x_n/s_n$ from Γ to Γ':
$$[\![\gamma]\!] : (p, \alpha') \mapsto \big(p, ([\![\Gamma'|s_1]\!]_{(p,\alpha')}, \ldots, [\![\Gamma'|s_n]\!]_{(p,\alpha')})\big) \qquad \text{for } (p, \alpha') \in [\![\Gamma']\!]$$
We write $I([\![\gamma]\!])$ for the induced natural transformation $I([\![\Gamma']\!]) \to I([\![\Gamma]\!])$.
- Basic types:
$$[\![\Gamma|a\ \gamma_0]\!] := [\![a]\!] \circ [\![\gamma_0]\!]$$
- Complex types:
$$\begin{aligned} [\![\Gamma|1]\!](p, \alpha) &:= \{\varnothing\} \\ [\![\Gamma|Id(s, s')]\!](p, \alpha) &:= \begin{cases} \{\varnothing\} & \text{if } [\![\Gamma|s]\!]_{(p,\alpha)} = [\![\Gamma|s']\!]_{(p,\alpha)} \\ \varnothing & \text{otherwise} \end{cases} \\ [\![\Gamma|\Sigma_{x:S}\,T]\!] &:= \mathcal{L}_{[\![\Gamma|S]\!]}[\![\Gamma, x : S|T]\!] \\ [\![\Gamma|\Pi_{x:S}\,T]\!] &:= \mathcal{R}_{[\![\Gamma|S]\!]}[\![\Gamma, x : S|T]\!] \end{aligned}$$
$[\![\Gamma|1]\!]$ and $[\![\Gamma|Id(s, s')]\!]$ are only specified for objects; their extension to morphisms is uniquely determined.

- Basic terms:

$$[\![\Gamma|c]\!]_{(p,\alpha)} := [\![c]\!]_p, \qquad [\![x_1 : S_1, \ldots, x_n : S_n | x_i]\!]_{(p,(a_1,\ldots,a_n))} := a_i$$

- Complex terms:

$$\begin{array}{ll} [\![\Gamma|*]\!]_{(p,\alpha)} & := \varnothing \\ [\![\Gamma|\mathit{refl}(s)]\!]_{(p,\alpha)} & := \varnothing \\ [\![\Gamma|\langle s, s'\rangle]\!]_{(p,\alpha)} & := ([\![\Gamma|s]\!]_{(p,\alpha)}, [\![\Gamma|s']\!]_{(p,\alpha)}) \\ [\![\Gamma|\pi_i(u)]\!]_{(p,\alpha)} & := a_i \quad \text{where } [\![\Gamma|u]\!]_{(p,\alpha)} = (a_1, a_2) \\ [\![\Gamma|\lambda_{x:S}\, t]\!] & := \mathrm{sp}([\![\Gamma, x : S|t]\!]) \\ [\![\Gamma|f\ s]\!] & := \mathrm{am}([\![\Gamma|f]\!]) * (\mathit{assoc} \circ F([\![\Gamma|s]\!])) \end{array}$$

  Here *assoc* maps $((p,\alpha),a)$ to $(p,(\alpha,a))$.

Since the same expression may have more than one well-formedness derivation, the well-definedness of Def. 5.2 must be proved in a joint induction with the proof of Thm. 7.1 below (see also [Str91]). And because of the use of substitution, e.g., for application of function terms, the induction must be intertwined with the proof of Thm. 6.1 as well.

Example 5.3 (Continuing Ex. 2.2)**.** A model of the signature *Cat* over an indexing poset P is the same thing as a functor from P into $\mathcal{CAT}$, the category of (small) categories. In more detail, assume a poset P and a functor $F : P \to \mathcal{CAT}$. Then we obtain a model of the signature *Cat* as follows:

- The underlying poset is P.
- $[\![Ob]\!]$ is the indexed set over P mapping
  - every $p \in P$ to the set of objects of $F(p)$,
  - every morphism $p \leq p'$ to the object-part of $F(p \leq p')$.
- $[\![x : Ob, y : Ob]\!]$ is a poset containing tuples $(p,(a,b))$ for $a, b \in F(p)$. We obtain $(p,(a,b)) \leq (p',(a',b'))$ iff $p \leq p'$ and $a' = F(p \leq p')(a)$ and $b' = F(p \leq p')(b)$. Then $[\![Mor]\!]$ is the indexed set over $[\![x : Ob, y : Ob]\!]$ mapping
  - every $(p,(a,b))$ to the set $\mathrm{Hom}_{F(p)}(a,b)$,
  - every $(p,(a,b)) \leq (p',(a',b'))$ to the morphism part of $F(p \leq p')$ restricted to a map from $\mathrm{Hom}_{F(p)}(a,b)$ to $\mathrm{Hom}_{F(p')}(a',b')$.
- Next we define $[\![id]\!] \in \mathrm{Elem}([\![\cdot | \Pi_{x:Ob}\, \mathit{Mor}\ x\ x]\!])$ as $\mathrm{sp}(e)$ (using Lem. 4.9) where $e \in \mathrm{Elem}([\![x : Ob | \mathit{Mor}\ x\ x]\!])$ is defined as follows. $[\![x : Ob | \mathit{Mor}\ x\ x]\!]$ maps (p,a) for $a \in [\![\cdot | Ob]\!](p)$ to the set $\mathrm{Hom}_{F(p)}(a,a)$, and we put $e_{(p,a)} := \mathrm{id}_a$.
  Because F is a functor, we have

$$[\![x : Ob | \mathit{Mor}\ x\ x]\!]((p,a) \leq (p',a'))(\mathrm{id}_a) = \mathrm{id}_{a'}.$$

  Therefore, e is indeed an indexed element.
- *comp* is interpreted as composition in $F(p)$ in the same manner as *id* applying Lem. 4.9 five times.
- The interpretations of the constants representing axioms such as *neutr* are uniquely determined. And they exist because all $F(p)$ are categories.

6. Substitution Lemma

Parallel to Lem. 2.3, we obtain the following central result about the semantics of substitutions. It expresses the coherence of our models.

Theorem 6.1 (Substitution)**.** *Assume* $\vdash_\Sigma \gamma : \Gamma \to \Gamma'$*. Then:*

$$\begin{array}{llll} \textit{if} & \vdash_\Sigma \delta : \Delta \to \Gamma & \textit{then} & [\![\gamma \circ \delta]\!] = [\![\delta]\!] \circ [\![\gamma]\!], \\ \textit{if} & \Gamma \vdash_\Sigma S : \mathtt{type} & \textit{then} & [\![\Gamma'|\gamma(S)]\!] = [\![\Gamma|S]\!] \circ [\![\gamma]\!], \\ \textit{if} & \Gamma \vdash_\Sigma s : S & \textit{then} & [\![\Gamma'|\gamma(s)]\!] = [\![\Gamma|s]\!] * [\![\gamma]\!]. \end{array}$$

Before we give the proof of Thm. 6.1, we establish some auxiliary results:

Lemma 6.2. *Assume* $\vdash_\Sigma \gamma : \Gamma \to \Gamma'$ *and* $\Gamma \vdash_\Sigma S : \mathtt{type}$ *and thus also*

$$\vdash_\Sigma \ \gamma, x/x \ : \ \Gamma, x{:}S \ \to \ \Gamma', x{:}\gamma(S) \ .$$

Furthermore, assume the induction hypothesis of Thm. 6.1 for the involved expressions. Then we have:

$$[\![\gamma, x/x]\!] = F(I([\![\gamma]\!]) \ltimes [\![\Gamma|S]\!]).$$

Proof. This follows by direct computation. □

Lemma 6.3. *Assume* $P|A|B$*,* $P|A \ltimes B|C$*,* $P|A'$*, a natural transformation* $g : A' \to A$*, and* $t \in \mathrm{Elem}(C)$*. Then for* $x' \in \int A'$*:*

$$\mathrm{sp}(t * F(g \ltimes B))_{x'} = \mathrm{sp}(t)_{F(g)(x')}.$$

Proof. This follows by direct computation. □

Proof of Thm. 6.1. The proofs of all subtheorems are intertwined in an induction on the typing derivations; in addition, the induction is intertwined with the proof of Thm. 7.1.

The case of an empty *substitution* δ is trivial. For the remaining cases, assume $\delta = x_1/s_1, \ldots, x_n/s_n$ and $(p, \alpha') \in [\![\Gamma']\!]$. Then applying the composition of substitutions, the semantics of substitutions, the induction hypothesis for terms, and the semantics of substitutions, respectively, yields:

$$\begin{aligned} &[\![\gamma \circ \delta]\!](p, \alpha') = [\![x_1/\gamma(s_1), \ldots, x_n/\gamma(s_n)]\!](p, \alpha') = \big(p, \big([\![\Gamma'|\gamma(s_i)]\!]_{(p,\alpha')}\big)_{i=1,\ldots,n}\big) \\ &= \big(p, \big([\![\Gamma|s_i]\!]_{[\![\gamma]\!](p,\alpha')}\big)_{i=1,\ldots,n}\big) = ([\![\delta]\!] \circ [\![\gamma]\!])(p, \alpha') \end{aligned}$$

The cases for *types* are as follows:

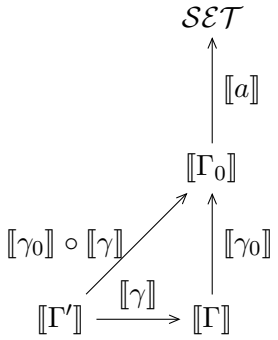
- $a\ \gamma_0$: Using the definition of substitution and the semantics of application, we obtain:

$$[\![\Gamma'|\gamma(a\ \gamma_0)]\!] = [\![\Gamma'|a\ (\gamma_0 \circ \gamma)]\!] = [\![a]\!] \circ [\![\gamma_0 \circ \gamma]\!]$$

  And similarly we obtain:

$$[\![\Gamma|a\ \gamma_0]\!] = [\![a]\!] \circ [\![\gamma_0]\!]$$

  Then the needed equality follows from the induction hypothesis for γ_0.



- 1: Trivial.
- $Id(s, s')$: This follows directly from the induction hypothesis for s and s'.
- $\Sigma_{x:S}\, T$: This follows directly by combining the induction hypothesis as well as Lem. 4.4 and 6.2.
- $\Pi_{x:S}\, T$: This follows directly by combining the induction hypothesis as well as Lem. 4.7 and 6.2.

For the cases of a *term* s, let us assume a fixed $(p, \alpha') \in [\![\Gamma']\!]$ and $(p, \alpha) := [\![\gamma]\!](p, \alpha')$. Then we need to show

$$[\![\Gamma'|\gamma(s)]\!]_{(p,a')} = [\![\Gamma|s]\!]_{(p,a)}.$$

- c: Clear because $\gamma(c) = c$.
- x: Assume x occurs in position i in Γ, and let x/s be in γ. Further, assume $\alpha' = (a'_1, \ldots, a'_n)$ and $\alpha = (a_1, \ldots, a_n)$. Then by the properties of substitutions: $[\![\Gamma'|\gamma(x)]\!]_{(p,\alpha')} = [\![\Gamma'|s]\!]_{(p,\alpha')} = a_i$. And that is equal to $[\![\Gamma|x]\!]_{(p,\alpha)}$.
- *refl*(s): Trivial.
- $*$: Trivial.
- $\langle s, s' \rangle$: Because $\gamma(\langle s, s' \rangle) = \langle \gamma(s), \gamma(s') \rangle$, this case follows immediately from the induction hypothesis.
- $\pi_i(u)$ for $i = 1, 2$: Because $\gamma(\pi_i(s)) = \pi_i(\gamma(s))$, this case follows immediately from the induction hypothesis.
- $\lambda_{x:S}\, t$: By the definition of substitution, the semantics of λ-abstraction, the induction hypothesis, and Lem. 6.2, respectively, we obtain:

$$\begin{aligned}[\![\Gamma'|\gamma(\lambda_{x:S}\, t)]\!] &= [\![\Gamma'|\lambda_{x:\gamma(S)}\, \gamma^x(t)]\!] = \mathrm{sp}([\![\Gamma', x\!:\!\gamma(S)|\gamma^x(t)]\!]) \\ &= \mathrm{sp}([\![\Gamma, x\!:\!S|t]\!] * [\![\gamma, x/x]\!]) \\ &= \mathrm{sp}([\![\Gamma, x\!:\!S|t]\!] * F(I([\![\gamma]\!]) \ltimes [\![\Gamma|S]\!])).\end{aligned}$$

  Furthermore, we have $[\![\Gamma|\lambda_{x:S}\, t]\!] = \mathrm{sp}([\![\Gamma, x\!:\!S|t]\!])$. Then the result follows by using Lem. 6.3 and $F(I([\![\gamma]\!])) = [\![\gamma]\!]$.
- $f\ s$: We evaluate both sides of the needed equation. Firstly, on the left-hand side, we obtain by the definition of substitution, the semantics of application, and the induction hypothesis, respectively:

$$\begin{aligned}[\![\Gamma'|\gamma(f\ s)]\!] = [\![\Gamma'|\gamma(f)\ \gamma(s)]\!] &= \mathrm{am}([\![\Gamma'|\gamma(f)]\!]) * (\mathit{assoc} \circ F([\![\Gamma'|\gamma(s)]\!])) \\ &= \mathrm{am}([\![\Gamma|f]\!] * [\![\gamma]\!]) * (\mathit{assoc} \circ F([\![\Gamma|s]\!] * [\![\gamma]\!])).\end{aligned}$$

  To compute the value at (p, α') of this indexed element, we first compute $([\![\Gamma|s]\!] * [\![\gamma]\!])_{(p,\alpha')}$, say we obtain b. Then we can compute $\mathrm{am}([\![\Gamma|f]\!] * [\![\gamma]\!])_{(p,(\alpha',b))}$. Using the notation from

Lem. 4.9, the left-hand side evaluates to

$$\big([\![\Gamma|f]\!] * [\![\gamma]\!]\big)^{(p,(\alpha',b))} = ([\![\Gamma|f]\!]_{(p,\alpha)})_{(p,(\varnothing,b))}.$$

Secondly, on the right-hand side, we have by the semantics of application:

$$[\![\Gamma|f\ s]\!] = \mathrm{am}([\![\Gamma|f]\!]) * (\mathit{assoc} \circ F([\![\Gamma|s]\!])).$$

When computing the value at (p,α) of this indexed element, we obtain in a first step $\mathrm{am}([\![\Gamma|f]\!])_{(p,(\alpha,b))}$. And evaluating further, this yields $([\![\Gamma|f]\!]_{(p,\alpha)})_{(p,(\varnothing,b))}$.

Thus, the equality holds as needed. □

7. Soundness

We have already mentioned the soundness result, which states that the interpretation takes the syntactic judgments for equality of terms and types to corresponding semantic judgments:

Theorem 7.1 (Soundness)**.** *Assume a signature Σ, and a context Γ. If $\Gamma \vdash_\Sigma S \equiv S'$ for two well-formed types S, S', then in every Σ-model:*

$$[\![\Gamma|S]\!] = [\![\Gamma|S']\!] \in \mathcal{SET}^{[\![\Gamma]\!]}.$$

And if $\Gamma \vdash_\Sigma s \equiv s'$ for two well-formed terms s, s' of type S, then in every Σ-model:

$$[\![\Gamma|s]\!] = [\![\Gamma|s']\!] \in \mathrm{Elem}([\![\Gamma|S]\!]).$$

Proof. The soundness is proved by induction over all derivations; the induction is intertwined with the proof of Thm. 6.1. An instructive example is the rule e_{typing}. Its soundness states the following: If $[\![\Gamma|s]\!] \in \mathrm{Elem}([\![\Gamma|S]\!])$ and $[\![\Gamma|s]\!] = [\![\Gamma|s']\!]$ and $[\![\Gamma|S]\!] = [\![\Gamma|S']\!]$, then also $[\![\Gamma|s']\!] \in \mathrm{Elem}([\![\Gamma|S']\!])$. And this clearly holds.

Among the remaining rules for terms, the soundness of some rules is an immediate consequence of the semantics. These are: all rules from Fig. 5 except for t_λ and t_{app}, and from Fig. 6 the rules $e_{Id(-,-)}$, $e_{id-uniq}$, e_*, $e_{\langle -,-\rangle}$, e_{π_1}, e_{π_2}, and e_{app}.

The soundness of the rules t_λ and t_{app} follows by applying the semantics and Lem. 4.9. That leaves the rules e_β and $e_{funcext}$, the soundness of which we will prove in detail.

For e_β, we interpret $(\lambda_{x:S}\, t)\ s$ by applying the definition:

$$\begin{aligned}[\![\Gamma|(\lambda_{x:S}\, t)\ s]\!] &= \mathrm{am}([\![\Gamma|\lambda_{x:S}\, t]\!]) * (\mathit{assoc} \circ F([\![\Gamma|s]\!])) \\ &= \mathrm{am}(\mathrm{sp}([\![\Gamma, x:S|t]\!])) * (\mathit{assoc} \circ F([\![\Gamma|s]\!]))\end{aligned}$$

$\mathrm{am}(\mathrm{sp}([\![\Gamma, x:S|t]\!]))$ is equal to $[\![\Gamma, x:S|t]\!]$ by Lem. 4.9. Furthermore, we have $t[x/s] = \gamma(t)$ where $\gamma = \mathrm{id}_\Gamma, x/s$ is a substitution from $\Gamma, x:S$ to Γ. And interpreting γ yields $[\![\gamma]\!](p,\alpha) = (p, (\alpha, [\![\Gamma|s]\!]_{(p,\alpha)}))$, i.e., $[\![\gamma]\!] = \mathit{assoc} \circ F([\![\Gamma|s]\!])$. Therefore, using Thm. 6.1 for terms yields

$$[\![\Gamma|t[x/s]]\!] = [\![\Gamma, x:S|t]\!] * (\mathit{assoc} \circ F([\![\Gamma|s]\!])),$$

which concludes the soundness proof for e_β.

To understand the soundness of $e_{funcext}$, let us look at the interpretations of f in the contexts Γ and $\Gamma, y:S$:

$$\mathrm{am}([\![\Gamma|f]\!]) \in \mathrm{Elem}([\![\Gamma, x:S|T]\!]), \qquad \mathrm{am}([\![\Gamma, y:S|f]\!]) \in \mathrm{Elem}([\![\Gamma, y:S, x:S|T]\!]).$$

Let γ be the inclusion substitution from Γ to $\Gamma, y : S$. Then $[\![\gamma]\!]$ is the projection $[\![\Gamma, y : S]\!] \to [\![\Gamma]\!]$ mapping elements $(p, (\alpha, a))$ to (p, α). Applying Thm. 6.1 yields for arbitrary $(p, \alpha) \in [\![\Gamma]\!]$ and $a', a \in [\![\Gamma | S]\!](p, \alpha)$:

$$\mathrm{am}([\![\Gamma, y : S | f]\!])_{(p,(\alpha,a',a))} = \mathrm{am}([\![\Gamma | f]\!])_{(p,(\alpha,a))}.$$

And we have

$$[\![\Gamma, y : S | y]\!]_{(p,(\alpha,a'))} = a', \qquad \text{and} \qquad F([\![\Gamma, y : S | y]\!])(p, (\alpha, a')) = (p, (\alpha, a'), a').$$

Putting these together yields

$$\begin{aligned} [\![\Gamma, y : S | f\ y]\!]_{(p,(\alpha,a'))} &= \big(\mathrm{am}([\![\Gamma, y : S | f]\!]) * (\mathit{assoc} \circ F([\![\Gamma, y : S | y]\!]))\big)_{(p,(\alpha,a'))} \\ &= \mathrm{am}([\![\Gamma, y : S | f]\!])_{(p,(\alpha,a',a'))} = \mathrm{am}([\![\Gamma | f]\!])_{(p,(\alpha,a'))} \end{aligned}$$

Therefore, the induction hypothesis applied to $\Gamma, y : S \vdash_\Sigma f\ y \equiv f'\ y$ yields

$$\mathrm{am}([\![\Gamma | f]\!]) = \mathrm{am}([\![\Gamma | f']\!]).$$

And then Lem. 4.9 yields

$$[\![\Gamma | f]\!] = [\![\Gamma | f']\!]$$

concluding the soundness proof for e_{funcext}.

Regarding the rules for types in Fig. 4 and Fig. 7, the soundness proofs are straightforward. □

8. Completeness

According to the propositions-as-types interpretation — also known as the Curry-Howard correspondence — a type S holds in a model if its interpretation $[\![S]\!]$ is inhabited, i.e., the indexed set $[\![S]\!]$ has an indexed element. A type is valid if it holds in all models. Then soundness implies: If there is a term s of type S in context Γ, then in every Σ-model there is an indexed element of $[\![\Gamma | S]\!]$, namely $[\![\Gamma | s]\!]$. The converse is completeness: A type that has an indexed element in every model is inhabited. Observe that the presence of (extensional) identity types then implies also the completeness of the equational term calculus because two terms are equal iff the corresponding identity type is inhabited.

The basic idea of the proof of completeness is to build the syntactic category, and then to construct a model out of it using categorical embedding theorems.

Definition 8.1. A functor $F : \mathcal{C} \to \mathcal{D}$ is called LCC if $\mathcal{C}$ is LCC and if F preserves that structure, i.e., F maps terminal object, products and exponentials in all slices $\mathcal{C}/A$ to corresponding structures in $\mathcal{D}/F(A)$. An LCC functor is called an LCC embedding if it is injective on objects, full, and faithful.

We make use of a theorem from topos theory due to Butz and Moerdijk ([BM99]) to establish the following central lemma.

Lemma 8.2. *For every LCC category $\mathcal{C}$, there is a poset P and an LCC embedding $E : \mathcal{C} \to \mathcal{SET}^P$.*

Proof. Clearly, the composition of LCC embeddings is an LCC embedding. We obtain $E : \mathcal{C} \to \mathcal{SET}^P$ as a composite $E_3 \circ E_2 \circ E_1$. Here $E_1 : \mathcal{C} \to \mathcal{SET}^{\mathcal{C}^{op}}$ is the Yoneda embedding, which maps $A \in |\mathcal{C}|$ to $\mathrm{Hom}(-, A)$. This is well-known to be an LCC embedding. E_2 maps a presheaf on $\mathcal{C}$ to a sheaf on a topological space S. E_2 is the inverse image part of the spatial cover of the topos $\mathcal{SET}^{\mathcal{C}^{op}}$ of presheaves on $\mathcal{C}$. This construction rests on a general topos-theoretical result established in [BM99], and we refer to [Awo00] for the details of the construction of S, the definition of E_2, and the proof that E_2 is an LCC embedding. Finally $E_3 : sh(S) \to \mathcal{SET}^{O(S)^{op}}$ includes a sheaf on S into the category of presheaves on the poset $O(S)$ of open sets of S. That E_3 is an LCC embedding, can be verified directly. Finally, we put $P := O(S)^{op}$ so that E becomes an LCC embedding into $\mathcal{SET}^P$. □

Definition 8.3 (Term-Generated)**.** A Σ-model I is called term-generated if for all closed Σ-types S and every indexed element $e \in \mathrm{Elem}([\![\cdot|S]\!]^I)$, there is a Σ-term s of type S such that $[\![\cdot|s]\!]^I = e$.

Theorem 8.4 (Model Existence)**.** *For every signature Σ, there is a term-generated model I such that for all types $\Gamma \vdash_\Sigma S : \mathtt{type}$*

$$\mathrm{Elem}([\![\Gamma|S]\!]^I) \neq \varnothing \quad \text{iff} \quad \Gamma \vdash_\Sigma s : S \text{ for some } s, \tag{8.1}$$

and for all such terms $\Gamma \vdash_\Sigma s : S$ and $\Gamma \vdash_\Sigma s' : S$

$$[\![\Gamma|s]\!]^I = [\![\Gamma|s']\!]^I \quad \text{iff} \quad \Gamma \vdash_\Sigma s \equiv s'. \tag{8.2}$$

Proof. It is well known how to construct the syntactic category $\mathcal{C}$ from Σ and Γ ([See84]). The objects of $\mathcal{C}$ are given by the set of all types S such that $\vdash_\Sigma S : \mathtt{type}$ modulo the equivalence relation $\vdash_\Sigma S \equiv S'$. We will write $[S]$ for the equivalence class of S.

The $\mathcal{C}$-morphisms from $[S]$ to $[S']$ are given by the terms f such that $\vdash_\Sigma f : S \to S'$ modulo the equivalence relation $\vdash_\Sigma f \equiv f'$. We will write $[f]$ for the equivalence class of f.

It is straightforward to check that $\mathcal{C}$ is LCC (see, e.g., [See84]). For example, the exponential $f_2^{f_1}$ of two objects $\vdash_\Sigma f_1 : S_1 \to S$ and $\vdash_\Sigma f_2 : S_2 \to S$ in a slice $\mathcal{C}/[S]$ is given by

$$\lambda_{u:U}\, \pi_1(u) \quad \text{where} \quad U := \Sigma_{x:S} \left(\Sigma_{y_1:S_1} Id(x, f_1\ y_1) \to \Sigma_{y_2:S_2} Id(x, f_2\ y_2)\right).$$

By Lem. 8.2, there are a poset P and an LCC embedding $E : \mathcal{C} \to \mathcal{SET}^P$. From those, we construct the needed model I over P. Essentially, I arises by interpreting every term or type as its image under E.

Firstly, assume a declaration $c : S$ in Σ. Since $\mathcal{C}$ only uses types and function terms, E cannot in general be applied to c. But using the type 1, every term c of type S can be seen as the function term $\lambda_{x:1}\, c$ of type $1 \to S$. Therefore, we define $E'(c) := E([\lambda_{x:1}\, c])$, which is an indexed element of $E([1 \to S])$. Since $\mathrm{Elem}(E([1 \to S]))$ and $\mathrm{Elem}(E([S]))$ are in bijection, $E'(c)$ induces an indexed element of $E([S])$, which we use to define $[\![c]\!]^I$.

Secondly, assume a declaration $a : (\Gamma_0)\mathtt{type}$ in Σ for $\Gamma_0 = x_1 : S_1, \ldots, x_n : S_n$. $[\![a]\!]^I$ must be an indexed set over $[\![\Gamma_0]\!]^I$. For the same reason as above, E cannot be applied directly to a. Instead, we use the type $U := \Sigma_{x_1:S_1} \ldots \Sigma_{x_n:S_n} (a\ \mathrm{id}_{\Gamma_0})$. The fibration $F(E([U])) : \int_P E(U) \to P$ factors canonically through $[\![\Gamma_0]\!]^I$, from which we obtain the needed indexed set $[\![a]\!]^I$.

That I is term-generated now follows directly from the fullness of E. Finally, the required property (8.1) clearly follows from I being term-generated, and (8.2) from the fact that E is faithful. □

The fact that the model I just constructed is term-generated can be interpreted as functional completeness of the semantics: If a natural transformation of a certain type exists in every model, then it is syntactically definable. In more detail, let I be the model constructed in Thm. 8.4, and assume a natural transformation $\eta : [\![\cdot|S]\!]^I \to [\![\cdot|S']\!]^I$ for some Σ-types S and S'. Then there exists a Σ-term f of type $S \to S'$ such that η arises from $[\![\cdot|f]\!]^I$ as follows. Put $\eta' := \mathrm{am}([\![\cdot|f]\!]^I) \in \mathrm{Elem}([\![x:S|S']\!]^I)$. Then η' maps pairs (p, a) to elements of $[\![x:S|S']\!]^I(p, a) = [\![\cdot|S']\!]^I(p)$ for $a \in [\![\cdot|S]\!]^I(p)$. Then we obtain η as $\eta_p : a \mapsto \eta'(p, a)$.

Theorem 8.5 (Completeness)**.** *For every signature Σ and any type $\Gamma \vdash_\Sigma S$: `type`, the following hold:*

(1) *If in every Σ-model I we have*

$$\mathrm{Elem}([\![\Gamma|S]\!]^I) \neq \varnothing,$$

then there is a term s with

$$\Gamma \vdash_\Sigma s : S.$$

(2) *For all terms $\Gamma \vdash_\Sigma s : S$ and $\Gamma \vdash_\Sigma s' : S$, if $[\![\Gamma|s]\!]^I = [\![\Gamma|s']\!]^I$ holds for all Σ-models I, then $\Gamma \vdash_\Sigma s \equiv s'$.*

Proof. This follows immediately from Thm. 8.4, considering the term-generated model constructed there. □

Finally, observe that in the presence of extensional identity types, statement (1) of Thm. 8.5 already implies statement (2): For all well-formed terms s, s' of type S, if $[\![\Gamma|s]\!] = [\![\Gamma|s']\!]$ in all Σ-models, then $[\![\Gamma|Id(s, s')]\!]$ always has an element, and so there must be a term $\Gamma \vdash_\Sigma t : Id(s, s')$, whence $\Gamma \vdash_\Sigma s \equiv s'$. An analogous result for types is more complicated and remains future work.

9. Conclusion and Future Work

We have presented a concrete and intuitive semantics for MLTT in terms of indexed sets on posets. And we have shown soundness and completeness. Our semantics is essentially that proposed by Lawvere in [Law69] in the hyperdoctrine of posets, fibrations, and indexed sets on posets, but we have made particular choices for which the models are coherent. Our models use standard function spaces, and substitution has a very simple interpretation as composition. The same holds in the simply-typed case, which makes our models an interesting alternative to (non-standard) Henkin models. In both cases, we strengthen the existing completeness results by restricting the class of models.

We assume that the completeness result can still be strengthened somewhat further, e.g., to permit equality axioms between types. In addition, it is an open problem to find an elementary completeness proof, i.e., one that does not rely on topos-theoretical results.

References

[All87] S. Allen. A Non-Type-Theoretic Definition of Martin-Löf's Types. In D. Gries, editor, *Proceedings of the Second Annual IEEE Symp. on Logic in Computer Science, LICS 1987*, pages 215–221. IEEE Computer Society Press, 1987.

[AR09] S. Awodey and F. Rabe. Kripke Semantics for Martin-Löf's Extensional Type Theory. In P. Curien, editor, *Typed Lambda Calculi and Applications (TLCA)*, volume 5608 of *Lecture Notes in Computer Science*, pages 249–263. Springer, 2009.

[Awo00] S. Awodey. Topological representation of the lambda-calculus. *Mathematical Structures in Computer Science*, 10(1):81–96, 2000.
[Bar92] H. Barendregt. Lambda calculi with types. In S. Abramsky, D. Gabbay, and T. Maibaum, editors, *Handbook of Logic in Computer Science*, volume 2. Oxford University Press, 1992.
[BM99] C. Butz and I. Moerdijk. Topological representation of sheaf cohomology of sites. *Compositio Mathematica*, 2(118):217–233, 1999.
[Car86] J. Cartmell. Generalized algebraic theories and contextual category. *Annals of Pure and Applied Logic*, 32:209–243, 1986.
[CF58] H. Curry and R. Feys. *Combinatory Logic*. North-Holland, Amsterdam, 1958.
[Chu40] A. Church. A Formulation of the Simple Theory of Types. *Journal of Symbolic Logic*, 5(1):56–68, 1940.
[Cur89] P. Curien. Alpha-Conversion, Conditions on Variables and Categorical Logic. *Studia Logica*, 48(3):319–360, 1989.
[Fri75] H. Friedman. Equality Between Functionals. In R. Parikh, editor, *Logic Colloquium*, pages 22–37. Springer, 1975.
[Hen50] L. Henkin. Completeness in the Theory of Types. *Journal of Symbolic Logic*, 15(2):81–91, 1950.
[HHP93] R. Harper, F. Honsell, and G. Plotkin. A framework for defining logics. *Journal of the Association for Computing Machinery*, 40(1):143–184, 1993.
[Hof94] M. Hofmann. On the Interpretation of Type Theory in Locally Cartesian Closed Categories. In *CSL*, pages 427–441. Springer, 1994.
[Hof97] M. Hofmann. Syntax and Semantics of Dependent Types. In A. Pitts and P. Dybjer, editors, *Semantics and Logic of Computation*, pages 79–130. Cambridge University Press, 1997.
[How80] W. Howard. The formulas-as-types notion of construction. In *To H.B. Curry: Essays on Combinatory Logic, Lambda-Calculus and Formalism*, pages 479–490. Academic Press, 1980.
[Jac90] B. Jacobs. *Categorical Type Theory*. PhD thesis, Catholic University of the Netherlands, 1990.
[Jac99] B. Jacobs. *Categorical Logic and Type Theory*. Elsevier, 1999.
[Joh02] P. Johnstone. *Sketches of an Elephant: A Topos Theory Compendium*. Oxford Science Publications, 2002.
[Kri65] S. Kripke. Semantical Analysis of Intuitionistic Logic I. In J. Crossley and M. Dummett, editors, *Formal Systems and Recursive Functions*, pages 92–130. North-Holland, 1965.
[Law69] W. Lawvere. Adjointness in Foundations. *Dialectica*, 23(3–4):281–296, 1969.
[Lip92] J. Lipton. Kripke Semantics for Dependent Type Theory and Realizability Interpretations. In J. Myers and M. O'Donnell, editors, *Constructivity in Computer Science, Summer Symposium*, pages 22–32. Springer, 1992.
[Mac98] S. Mac Lane. *Categories for the working mathematician*. Springer, 1998.
[ML84] P. Martin-Löf. *Intuitionistic Type Theory*. Bibliopolis, 1984.
[MM91] J. Mitchell and E. Moggi. Kripke-style Models for Typed Lambda Calculus. *Annals of Pure and Applied Logic*, 51(1–2):99–124, 1991.
[MM92] S. Mac Lane and I. Moerdijk. *Sheaves in geometry and logic*. Lecture Notes in Mathematics. Springer, 1992.
[MS89] J. Mitchell and P. Scott. Typed lambda calculus and cartesian closed categories. In *Categories in Computer Science and Logic*, volume 92 of *Contemporary Mathematics*, pages 301–316. Amer. Math. Society, 1989.
[Pit00] A. Pitts. Categorical Logic. In S. Abramsky, D. Gabbay, and T. Maibaum, editors, *Handbook of Logic in Computer Science, Volume 5. Algebraic and Logical Structures*, chapter 2, pages 39–128. Oxford University Press, 2000.
[Rab08] F. Rabe. *Representing Logics and Logic Translations*. PhD thesis, Jacobs University Bremen, 2008. see `http://kwarc.info/frabe/Research/phdthesis.pdf`.
[See84] R. Seely. Locally cartesian closed categories and type theory. *Math. Proc. Cambridge Philos. Soc.*, 95:33–48, 1984.
[Sim95] A. Simpson. Categorical completeness results for the simply-typed lambda-calculus. In M. Dezani-Ciancaglini and G. Plotkin, editor, *Typed Lambda Calculi and Applications*, pages 414–427, 1995.
[Str91] T. Streicher. *Semantics of Type Theory*. Springer-Verlag, 1991.