# Modules as dependently typed records

Thierry Coquand, Randy Pollack and Makoto Takeyama

August 16, 2002 (1033)

## Introduction

We suggest a general model construction for type theory. It gives a natural model of subtyping and can be used to design type systems that satisfy by construction decidability and normalisation property. We believe that our proof of decidability, based on simple semantical ideas, is simpler than a syntactical argument.

We can describe quickly the main feature of our model. It is an usual PER model, thus to any type $A$ we associate one partial equivalence relation $\sim_A$ on the set of untyped $\lambda$-terms, and the associate computability predicate $\phi_A(M)$ meaning $M \sim_A M$. What may be new here is the connection between $\sim_A$ and a general form of $\eta$-expansion functions $\eta\ A$: first if $M$ is computable at type $A$ then $\eta\ A\ M$ is also computable, next if $M_1, M_2$ are computable at type $A$ then $M_1 \sim_A M_2$ iff $\eta\ A\ M_1$ and $\eta\ A\ M_2$ are convertible (where convertibility means usual $\beta$-conversion). To this we can add the new fact that if $M$ is computable at type $A$, we can only ensure that $\eta\ A\ M$ is normalisable (and $M$ itself may actually be not normalisable in general) and that a variable $x$ may not be computable at type $A$ (but its $\eta$-expanded form $\eta\ A\ x$ will be computable at type $A$).

For presenting our model, we start first with a minimal version with only dependent products. This simple model gives already decidability and normalisation for Martin-Löf logical framework with $\eta$-conversion. We explain then how to add dependent sums and a type unit, with corresponding surjective pairing rules. In this version we have a first form of subtyping, reminiscent of Automath's telescopes. The last version contains records with manifest fields.

## 1 Logical framework

### 1.1 Values

The values are usual *untyped* $\lambda$-terms. Application is written as in functional programming $M_1\ M_2$ and we write $M_1\ M_2\ M_3$ for $(M_1\ M_2)\ M_3$. We define as usual $I = \lambda x.x$. The values are given by the following syntax

$$M ::= x \mid M\ M \mid \lambda x.M$$

We consider the new syntactic categories of type and (unary) type family

$$A ::= El\ M \mid \Pi\ A\ F \mid\ *\ \mid F\ M \qquad\qquad F ::= \lambda x.A$$

and the categories of kinds and (unary) kind family

$$K ::= \Box \mid \Pi\ A\ F \mid F\ M \qquad F ::= \lambda x.K$$

We may write $(\Pi x{:}A)B$ for $\Pi\ A\ (\lambda x.B)$ and $(\Pi x{:}A)K$ for $\Pi\ A\ (\lambda x.K)$. We shall need also the category of general type families

$$T ::= \lambda x.T \mid T\ M \mid A$$

Let $\Lambda$ be the set of all values, not necessarily closed. The equality on values is *$\beta$-conversion.* We define the element $\eta\ A \in \Lambda$ by induction on $A$

$$\eta\ * = I \qquad \eta\ (El\ M) = I \qquad \eta\ (\Pi\ A\ F) = \lambda u.\lambda x.\eta\ (F\ (\eta\ A\ x))\ (u\ (\eta\ A\ x))$$

We write $A_1 \to A_2$ for $\Pi\ A_1\ (\lambda x.A_2)$ whenever $x$ is not free in $A_2$. The *base* types are types $A$ such that $\eta\ A = I$, so are the values of the form $*$ or $El\ M$. The term $\eta\ A$ represents $\eta$-expansion at type $A$ and $\eta\ A\ M$ is the *incarnation* of $M$ at type $A$. If for instance $A$ is a base type, we have

$$\eta\ ((A \to A) \to A) = \lambda u.\lambda y.u\ (\lambda x.y\ x)$$

We define as usual normalisable terms, and we say that a term is *neutral* iff it is normalisable and of the form

$$\nu\ =\ x \mid \nu\ M$$

Thus a neutral term $\nu$ is a term normalisable and non canonical. We emphasise that we do not have $\eta$-conversion at the level of values, though $\eta$-conversion is valid for the corresponding type system.

A *per* (partial equivalence relation) on $\Lambda$ is a relation which is symmetric and transitive. The equality on pers is extensional equality, i.e. $R = S$ iff $M_1\ R\ M_2$ is equivalent to $M_1\ S\ M_2$. We have a special per $\sim_N$ where $M_1 \sim_N M_2$ means that $M_1, M_2$ are neutral and equal.

## 1.2 Type and Kind

We define inductively a subset **Type** of the set of types. Simultaneously we define for each $A \in$ **Type** a per $\sim_A$. We may write $M_1 = M_2 : A$ for $M_1 \sim_A M_2$. If $A \in$ **Type** we write $F \in Fam(A)$ to mean that $M_1 = M_2 : A$ implies $F\ M_1 \in$ **Type** and $F\ M_2 \in$ **Type** and $\sim_{F\ M_1} = \sim_{F\ M_2}$. If $A_1, A_2 \in$ **Type** we say that $A_1$ is a *subtype* of $A_2$ iff $\sim_{A_1} \subseteq \sim_{A_2}$. The clauses are as follow

- $* \in$ **Type** and $\sim_*$ is $\sim_N$
- $El\ M \in$ **Type** whenever $M$ is neutral and $\sim_{El\ M}$ is $\sim_N$
- $\Pi\ A\ F \in$ **Type** whenever $A \in$ **Type** and $F \in Fam(A)$; in this case $M_1 = M_2 : \Pi\ A\ F$ means that $N_1 = N_2 : A$ implies $M_1\ N_1 = M_2\ N_2 : F\ N_1$

In the last clause we could write $M_1\ N_1 = M_2\ N_2 : F\ N_2$ instead since $F \in Fam(A)$.
We let $M : A$ to mean $M = M : A$.

We can now define inductively a subset **Kind** of the set of all kinds. Simultaneously we define for each $K \in$ **Kind** a per $\sim_K$ on the set of all type families. The clauses are as follow

- $\Box \in$ **Kind** and $F_1 \sim_\Box F_2$ iff $F_1, F_2 \in$ **Type** and $\sim_{F_1} = \sim_{F_2}$
- $\Pi\ A\ F \in$ **Kind** whenever $A \in$ **Type** and $M_1 = M_2{:}A$ implies $F\ M_1, F\ M_2 \in$ **Kind** and $\sim_{F\ M_1} = \sim_{F\ M_2}$; in this case $F_1 \sim_{\Pi\ a\ A} F_2$ means that $M_1 = M_2{:}A$ implies $F_1\ M_1 \sim_{F\ M_1} F_2\ M_2$

### 1.3 Incarnation and computability

**Theorem:** If $A \in \mathbf{Type}$ then

1. $\eta\ A\ \nu : A$ if $\nu$ is neutral
2. if $M : A$ then $\eta\ A\ M$ is normalisable
3. if $M : A$ then $M = \eta\ A\ M : A$
4. $M_1 = M_2 : A$ iff $M_1 : A$ and $M_2 : A$ and $\eta\ A\ M_1 = \eta\ A\ M_2$

**Proof:** The theorem is proved by induction on the proof that $A \in \mathbf{Type}$. This is direct if $A$ is a base type, since in this case $\eta\ A$ is $I$ and $\sim_A$ is $\sim_N$.

Consider now a type of the form $\Pi\ A\ F$. We have then

$$\eta\ (\Pi\ A\ F)\ M = \lambda x.\eta\ (F\ (\eta\ A\ x))\ (M\ (\eta\ A\ x))$$

1. Assume that $M$ is neutral, and that $M_1 = M_2 : a$. By induction hypothesis we have $\eta\ A\ M_1 = \eta\ A\ M_2$. Also $\eta\ A\ M_1 : a$ and $\eta\ A\ M_1$ is normalisable. So $t\ (\eta\ A\ M_1)$ is neutral and by induction hypothesis

$$\eta\ (F\ (\eta\ A\ M_1))\ (M\ (\eta\ A\ M_1)) : F\ (\eta\ A\ M_1)$$

Since $\sim_{F\ (\eta\ A\ M_1)} = \sim_{F\ M_1}$ it follows that we have $\eta\ (\Pi\ A\ F)\ M\ M_1 = \eta\ (\Pi\ A\ F)\ M\ M_2$ as expected.

2. Assume $M : \Pi\ A\ F$ then $\eta\ A\ x{:}A$ by induction hypothesis and hence $M\ (\eta\ A\ x) : F\ (\eta\ A\ x)$. So, by induction hypothesis

$$\eta\ (F\ (\eta\ A\ x))\ (M\ (\eta\ A\ x))$$

is normalisable and hence $\eta\ (\Pi\ A\ F)\ M$ is normalisable.

3. Assume $M : \Pi\ A\ F$ and $M_1 = M_2 : a$. We have $M\ M_1 : F\ M_1$ and

$$\sim_{F\ M_1} = \sim_{F\ M_2} = \sim_{F\ (\eta\ A\ M_2)}$$

By induction hypothesis

$$M\ (\eta\ A\ M_2) = \eta\ (F\ (\eta\ A\ M_2))\ (M\ (\eta\ A\ M_2)) : F\ (\eta\ A\ M_2)$$

and since $M\ M_1 = M\ (\eta\ A\ M_2) : F\ M_1$ we have

$$M\ M_1 = (\eta\ (\Pi\ A\ F)\ M)\ M_2 : F\ M_1$$

as desired.

4. If $M_1 = M_2 : \Pi\ A\ F$ we have by induction $\eta\ A\ x{:}A$ and hence $M_1\ (\eta\ A\ x) = M_2\ (\eta\ A\ x) : F\ (\eta\ A\ x)$. Hence, by induction

$$\eta\ (F\ (\eta\ A\ x))\ (M_1\ (\eta\ A\ x)) = \eta\ (F\ (\eta\ A\ x))\ (M_2\ (\eta\ A\ x))$$

which implies $\eta\ (\Pi\ A\ F)\ M_1 = \eta\ (\Pi\ A\ F)\ M_2$ as expected.

If we have $M_1 : \Pi\ A\ F,\ M_2 : \Pi\ A\ F$ and $\eta\ (\Pi\ A\ F)\ M_1 = \eta\ (\Pi\ A\ F)\ M_2$, and if $N_1 = N_2 : A$ then we have $M_1\ (\eta\ A\ N_1) = M_1\ N_1 : F\ N_1$, and $M_2\ (\eta\ A\ N_2) = M_2\ N_2 : F\ N_1$ and by induction $\eta\ A\ N_1 = \eta\ A\ N_2$. Since we have

$$\eta\ (F\ (\eta\ A\ x))\ (M_1\ (\eta\ A\ x)) = \eta\ (F\ (\eta\ A\ x))\ (M_2\ (\eta\ A\ x))$$

we have also

$$\eta\ (F\ (\eta\ A\ N_1))\ (M_1\ (\eta\ A\ N_1)) = \eta\ (F\ (\eta\ A\ N_1))\ (M_2\ (\eta\ A\ N_2))$$

which implies by induction

$$M_1\ (\eta\ A\ N_1) = M_2\ (\eta\ A\ N_2) : F\ (\eta\ A\ N_1)$$

and we conclude

$$M_1\ N_1 = M_2\ N_2 : F\ N_1$$

as desired.

**Corollary:** If $A \in \mathbf{Type}$ and $M_1 : A, M_2 : A$ then the relation $M_1 = M_2 : A$ is decidable.

**Proof:** By the theorem, $M_1 = M_2 : A$ is equivalent to

$$\eta\ A\ M_1 = \eta\ A\ M_2$$

and $\eta\ A\ M_1, \eta\ A\ M_2$ are normalisable, hence the result.

## 1.4 Hypothetical Judgements

An environment $\rho, \ldots$ is a function from identifiers (names) to $\lambda$-terms. We let $\rho_0$ be the identity environment $\rho_0(x) = x$ and, if $\rho$ is an environment $\rho, x = M$ is the *update* of $\rho$ defined by $(\rho, x = M)(x) = M$ and $(\rho, x = M)(y) = \rho(y)$ for $y \neq x$. If $\rho$ is an environment we write $M\rho$ the substitution of $\rho(x)$ for $x$ in $M$.

We consider now contexts

$$C ::= () \mid C, x{:}A \mid C, x{:}A = M$$

A context is a finite object of the form

$$C = x_1 : A_1, x_2 : A_2 = M, x_3 : A_3$$

The identifiers $x_1, x_2, x_3$ are then *declared* in $C$. We assume that one forming $C' = C, x{:}A$ and $C' = C, x{:}A = M$ the identifier $x$ has not been declared yet in $C$. The new context $C'$ is then said to *extend directly* $C$ and we say that a context *extends* another one if it can be obtained by a sequence of direct extension.

We define now what are *valid* contexts, and to each valid context $C$ we associate a per $\sim_C$ on the set of environments. We shall write $\rho_1 = \rho_2{:}C$ for $\rho_1 \sim_C \rho_2$ and $\rho : C$ for $\rho = \rho : C$. If $C$ is a valid context, we let $A$ **type** $[C]$ to mean that if $\rho_1 = \rho_2{:}C$ then $A\rho_1,\ A\rho_2$ are in **Type** and the corresponding pers are equivalent. Also we let $M_1 = M_2 : A\ [C]$ to mean that if $\rho_1 = \rho_2{:}C$ then $M_1\rho_1 = M_2\rho_2 : A\rho_1$. We write $M : A\ [C]$ for $M = M : A\ [C]$. For forming a valid context $C, x{:}A$ (resp. $C, x{:}A = M$) from a given context $C$ we require that $x$ is not yet declared in $C$ and that $A$ **type** $[C]$ (resp. $M : A\ [C]$.)

We let also $A_1 \subseteq A_2\ [C]$ to mean that if $\rho_1 = \rho_2{:}C$ then $A\rho_1,\ A\rho_2$ are in **Type** and $A\rho_1$ is a subtype of $A\rho_2$.

Notice that this can be seen as a direct formalisation of the notion of hypothetical judgement as described in [2].

The following theorem has a direct proof.

**Theorem:** We have

- if $A$ **type** $[C]$ then $x{:}A\ [C, x{:}A]$
- if $\rho_1 = \rho_2{:}C$ and $C'$ extends $C$ then $\rho_1 = \rho_2{:}C'$
- if $M_1{:}\Pi\ A\ F\ [C]$ and $M_2{:}A\ [C]$ then $M_1\ M_2{:}F\ M_2\ [C]$
- if $B$ **type** $[C, x{:}A]$ then $(\Pi\ x{:}A)B$ **type** $[C]$
- if $M{:}B\ [C, x{:}A]$ then $\lambda x.M{:}(\Pi\ x{:}A)B\ [C]$
- if $A_2 \subseteq A_1\ [C]$ and $F_1\ x \subseteq F_2\ x\ [C, x{:}A_2]$ then $\Pi\ A_1\ F_1 \subseteq \Pi\ A_2\ F_2\ [C]$

We shall need also the judgements $K$ **kind** $[C]$ and $T : K\ [C]$ that are defined in the expected way.

**Theorem:**

- if $K$ **kind** $[C, x{:}A]$ then $(\Pi x{:}A)K$ **kind** $[C]$
- if $T{:}\Pi\ A\ F\ [C]$ then $M{:}A\ [C]$ then $T\ M{:}F\ M\ [C]$
- if $T{:}K\ [C, x{:}A]$ then $\lambda x.T{:}(\Pi x{:}A)K\ [C]$

### 1.5 Decidability

We define $\eta\ C\ \rho$ for each context $C$ and environment $\rho$.

- $\eta\ ()\ \rho = \rho$
- $\eta\ (C, x{:}A)\ \rho = (\eta\ C\ \rho, x = \eta\ A(\eta\ C\ \rho)\ \rho(x))$
- $\eta\ (C, x{:}A = M)\ \rho = (\eta\ C\ \rho, x = \eta\ A(\eta\ C\ \rho)\ M(\eta\ C\ \rho))$

**Lemma:** If $\rho : C$ then $\rho = \eta\ C\ \rho : C$. Furthermore $\eta\ C\ \rho_0 : C$

We let $\rho_C$ be the environment $\eta\ C\ \rho_0$.

**Corollary:** If $M_1{:}A\ [C]$, $M_2{:}A\ [C]$ then $M_1 = M_2{:}A\ [C]$ is equivalent to $M_1\rho_C = M_2\rho_C : A\rho_C$ and hence is decidable.

## 2 Signatures with definitions

### 2.1 Signatures

We extend the terms with

$$M ::= (x = M, M) \mid M.x \mid ()$$

and $\beta$-reduction is extended with $(x = M, N).x \succ M$ and $(x = M, N).y \succ N.y$ for $y \neq x$.

The types are extended with

$$A ::= S$$

where $S$ is a *signature* defined as

$$S ::= 1 \mid x{:}A, G \mid x{:}A = M, S \mid G\ M \qquad G ::= \lambda x.S$$

We shall also consider

$$D ::= 1 \mid x{:}A = M, D$$

A signature is thus a syntactical object of the form

$$S = x_1 : A_1, \lambda x.x_2 : A_2 = M, x_3 : A_3, \lambda y.1$$

The identifiers $x_1, x_2, x_3$ are called the *labels* of the signature $S$. We shall only consider signatures with no duplicated labels.

We define a subset **Sig** of the set of signatures and for $S \in$ **Sig** a per $\sim_S$. Simultaneously we define the subset **Type** of the set of types as before in such a way that **Sig** $\subseteq$ **Type**.

- $1 \in$ **Sig** and $M_1 = M_2 : 1$ always holds
- $(x{:}A, G) \in$ **Sig** if $A \in$ **Type** and $N_1 = N_2 : A$ implies that $G\ N_1,\ G\ N_2$ are in **Sig** and the corresponding pers are equivalent. We define then $M_1 = M_2 : (x{:}A, G)$ to mean $M_1.x = M_2.x{:}A$ and $M_1 = M_2 : G\ (M_1.x)$
- $(x{:}A = M, S) \in$ **Sig** if $A \in$ **Type** and $M : A$ and $S \in$ **Sig**. We define then $M_1 = M_2 : (x{:}A = M, S)$ to mean $M_1.x = M : A,\ M_2.x = M : A$ and $M_1 = M_2 : S$

We extend the set of neutral terms: a term is neutral iff it is normalisable and of the form

$$\nu ::= x \mid \nu\ M \mid \nu.x$$

Because of the restriction on non duplication of labels, the following lemma holds.

**Lemma:** If $(x{:}A, G) \in$ **Sig** and $M_1 = M_2 : A$ and $N_1 = N_2 : G\ M_1$ then

$$(x = M_1, N_1) = (x = M_2, N_2) : (x{:}A, G)$$

We define

$$\eta\ 1 = \lambda u.() \qquad \eta\ (x{:}A, G) = \lambda u.(x = \eta\ A\ u.x, \eta\ (G\ (\eta\ A\ u.x)\ u)$$

$$\eta\ (x{:}A = N, S) = \lambda u.(x = \eta\ A\ N, \eta\ S\ u)$$

**Theorem:** If $A \in$ **Type** then

1. $\eta\ A\ \nu : A$ if $\nu$ is neutral
2. if $M : A$ then $\eta\ A\ M$ is normalisable
3. if $M : A$ then $M = \eta\ A\ M : A$
4. $M_1 = M_2 : A$ iff $M_1 : A$ and $M_2 : A$ and $\eta\ A\ M_1 = \eta\ A\ M_2$

### 2.2 Strengthening

We define $A[M]$ and $S[M]$ by case on $A$ and $S$.

- $*[M] = *, \quad (El\ N)[M] = El\ N$
- $((\Pi\ x{:}A)B)[M] = (\Pi\ x{:}A)B[M\ x]$
- $1[M] = 1$
- $(x{:}A, G)[M] = (x{:}A[M.x] = M.x, G\ (M.x)[M])$
- $(x{:}A = N, S)[M] = (x{:}A = N, S[M])$

Notice that we have $(A[M])\rho = A\rho[M\rho]$.

**Lemma:** If $M_1 = M_2 : A$ then $A[M_1], A[M_2] \in$ **Type** and the corresponding pers are equivalent. Furthermore $M_1 = M_2 : A[M_1]$.

Similarly we could define $A/M$ and $S/M$ with similar clauses except for

$$(x{:}A, G)/M = (x{:}A = M.x, G\ (M.x)/M)$$

The lemma holds then for $A/M$ instead of $A[M]$ and furthermore $A[M]$ is a subtype of $A/M$ if $M : A$

**Lemma:** We have $D \subseteq (x{:}A, G)$ if $x{:}B = N$ occurs in $D$ and $B \subseteq A$ and $D \subseteq F\ N$. Similarly $D \subseteq (x{:}A = M, S)$ if $x{:}B = N$ occurs in $D$ and $B \subseteq A$ and $N = M{:}A$ and $D \subseteq S$.

### 2.3 Hypothetical judgments

We extend directly the notion of context, valid context and hypothetical judgements

$$A\ \mathbf{type}\ [C] \quad A_1 \subseteq A_2\ [C] \quad M{:}A\ [C] \quad M_1 = M_2{:}A\ [C]$$

with the new signature types. We add also the new judgement $S$ **sig** $[C]$ to mean that if $\rho_1 = \rho_2{:}C$ then $S\rho_1, S\rho_2$ are in **Sig** and the corresponding pers are equivalent.

**Lemma:**

- If $A$ **type** $[C]$ then $x{:}A[x]\ [C, x{:}A]$
- If $A_1$ **type** $[C]$ and $A_2$ **type** $[C]$ and $A_1/x \subseteq A_2\ [C, x{:}A_1]$ then $A_1 \subseteq A_2\ [C]$
- If $(x{:}A, G)$ **sig** $[C]$ and $M{:}A\ [C]$ and $N{:}G\ M\ [C]$ then $(x = M, N){:}(x{:}A, G)\ [C]$
- If $M{:}A\ [C]$ and $S$ **sig** $[C, x{:}A = M]$ then $(x{:}A = M, S)$ **sig** $[C]$

## 3 Typechecking

### 3.1 Syntactical subtyping

We introduce inductively the relations $C \vdash S_1 \sqsubseteq S_2$ and $C \vdash A_1 \sqsubseteq A_2$.

$$\frac{M_1 = M_2{:}*\ \ [C]}{C \vdash El\ M_1 \sqsubseteq El\ M_2} \qquad \frac{}{C \vdash * \sqsubseteq *}$$

$$\frac{C \vdash A_2 \sqsubseteq A_1 \qquad C, x{:}A_1 \vdash F_1\ x \sqsubseteq F_2\ x}{C \vdash \Pi\ A_1\ F_1 \sqsubseteq \Pi\ A_2\ F_2}$$

$$\frac{}{C \vdash S \sqsubseteq 1}$$

$$\frac{C, x{:}S_1 \vdash S_1/x \sqsubseteq S_2}{C \vdash S_1 \sqsubseteq S_2}$$

$$\frac{x{:}B = N\ in\ D \qquad C \vdash B \sqsubseteq A \qquad C \vdash D \sqsubseteq G\ N}{C \vdash D \sqsubseteq (x{:}A, G)}$$

$$\frac{x{:}B = N\ in\ D \qquad C \vdash B \sqsubseteq A \qquad N = M{:}A\ [C] \qquad C \vdash D \sqsubseteq S}{C \vdash D \sqsubseteq (x{:}A = M, G)}$$

**Theorem:** If $A_1$ **type** $[C]$ and $A_2$ **type** $[C]$ then $C \vdash A_1 \sqsubseteq A_2$ is decidable. Furthermore $C \vdash A_1 \sqsubseteq A_2$ implies that $A_1 \subseteq A_2\ [C]$.

### 3.2 Syntax

We write $a, b, t, e, u \ldots$ the syntactical terms. The syntax is

$$\begin{array}{lll}
e, a & ::= & e\ e \mid x \mid s \mid [x = e]e \mid [x : a]e \mid e.x \mid () \mid (x = e, e) \mid \\
 & & El\ e \mid * \mid (x : a) \to a \mid a \to a \mid <\Gamma> \\
B & ::= & () \mid B, x : a \mid B, x = e \\
\Gamma & ::= & () \mid [x : a]\Gamma \mid [x = e]\Gamma
\end{array}$$

Here $B$ is a *book*, in Automath terminology. The type-checking algorithm checks the correctness of book $B$.

### 3.3 Typechecking

We define inductively the relations $B \vdash a \mapsto A$ and $B \vdash \Gamma \mapsto S$ and $B \vdash e \mapsto M : A$ and $B \mapsto C$ and $B \vdash t \mapsto T : K$. A *valid* book is a book such that $B \mapsto C$ for some (then uniquely determined) context $C$. We write $B \vdash A_1 \sqsubseteq A_2$ for $B \mapsto C$ and $C \vdash A_1 \sqsubseteq A_2$.

The rules for typechecking a book are the following.

$$\frac{}{() \mapsto ()} \qquad \frac{B \mapsto C \qquad B \vdash a \mapsto A}{B, x : a \mapsto C, x : A}$$

$$\frac{B \mapsto C \qquad B \vdash e \mapsto M : A}{B, x = e \mapsto C, x : A = M} \qquad \frac{B \mapsto C \qquad B \vdash t \mapsto T : K}{B, x = t \mapsto C}$$

Like for context we can define the *declared* names of a book $B$ and the *extension* relation on books. We consider only books with non duplicated declared names. The rules for accessing a variable are

$$\frac{B'\ extends\ B \qquad B \vdash x \mapsto J}{B' \vdash x \mapsto J}$$

$$\frac{B \vdash a \mapsto A}{B, x : a \vdash x \mapsto x{:}A[x]} \qquad \frac{B \vdash e \mapsto M : A}{B, x = e \vdash x \mapsto x{:}A[M]}$$

$$\frac{B \vdash t \mapsto T : K}{B, x = t \vdash x \mapsto T : K}$$

The remaining type-checking rules are:

$$\frac{}{B \vdash * \mapsto *} \qquad \frac{B \vdash e \mapsto M : *}{B \vdash El\ e \mapsto El\ M}$$

$$\frac{B \vdash a \mapsto A \qquad B, x{:}a \vdash a_1 \mapsto A_1}{B \vdash (x : a) \to a_1 \mapsto (\Pi x{:}A)A_1} \qquad \frac{B \vdash a \mapsto A \qquad B \vdash a_1 \mapsto A_1}{B \vdash a \to a_1 \mapsto (\Pi x{:}A)A_1}$$

$$\frac{B \vdash t \mapsto A : \Box}{B \vdash t \mapsto A}$$

$$\frac{B \vdash e \mapsto M{:}\Pi\ A\ F \qquad B \vdash e_1 \mapsto M_1{:}A_1 \qquad B \vdash A_1 \sqsubseteq A}{B \vdash e\ e_1 \mapsto M\ M_1{:}F\ M_1}$$

$$\frac{B \vdash t \mapsto T{:}\Pi\ A\ F \qquad B \vdash e_1 \mapsto M_1{:}A_1 \qquad B \vdash A_1 \sqsubseteq A}{B \vdash t\ e_1 \mapsto T\ M_1{:}F\ M_1}$$

$$\frac{B \vdash a \mapsto A \qquad B, x{:}a \vdash e \mapsto M{:}A_1}{B \vdash [x : a]e \mapsto \lambda x.M{:}(\Pi x{:}A)A_1} \qquad \frac{B \vdash a \mapsto A \qquad B, x{:}a \vdash t \mapsto T{:}K}{B \vdash [x : a]t \mapsto \lambda x.T{:}(\Pi x{:}A)K}$$

$$\frac{B \vdash e \mapsto M{:}A \qquad B, x = e \vdash e_1 \mapsto M_1{:}A_1}{B \vdash [x = e]e_1 \mapsto (M/x)M_1{:}(M/x)A_1}$$

$$\frac{}{B \vdash () \mapsto 1} \qquad \frac{B \vdash a \mapsto A \quad B, x{:}a \vdash \Gamma \mapsto S}{B \vdash [x : a]\Gamma \mapsto (x : A, \lambda x.S)}$$

$$\frac{B \vdash e \mapsto M : A \quad B, x = e \vdash \Gamma \mapsto S}{B \vdash [x = e]\Gamma \mapsto (x : A = M, (M/x)S)}$$

$$\frac{}{B \vdash () \mapsto () : 1} \qquad \frac{B \vdash e \mapsto M : A \quad B \vdash e_1 \mapsto M_1 : D}{B \vdash (x = e, e_1) \mapsto (x = M, M_1) : (x : A = M, D)}$$

$$\frac{B \vdash e \mapsto M{:}D \qquad x{:}A = N\ in\ D}{B \vdash e.x \mapsto N{:}A}$$

$$\frac{B \vdash \Gamma \mapsto S}{B \vdash <\Gamma> \mapsto S}$$

If $B \vdash e \mapsto M : A$ we say that $A$ is a *principal type* for $e$ in the book $B$. For instance the principal type of

$$[x : <l : *>]x$$

(in the empty book) is $(\Pi x{:}(l : *))(l : * = x.l)$

**Theorem:** The validity of a book is decidable.

If $B \mapsto C$ then $C$ is a valid context. If furthermore $B \vdash a \mapsto A$ then $A$ **type** $[C]$. If $B \vdash e \mapsto M : A$ then $M{:}A$ $[C]$. If $B \vdash t \mapsto T : K$ then $T{:}K$ $[C]$.

## References


[1] H. Barendregt. *The λ-calculus, syntax and semantics.* North-Holland, 1980.

[2] B. Nordström, K. Petersson, J. Smith. *Programming in Martin-Löf's Type Theory.* Oxford Science Publications, 1990.