Encoding Universe Polymorphism on the $\lambda\Pi/\equiv$-calculus

Thiago F Cesar

1 PTS

Terms

$$A, B, M, N ::= x \in \mathcal{X} \mid c \in \mathcal{C} \mid s \in \mathcal{S} \mid MN \mid \lambda x : A.M \mid \Pi x : A.B \mid$$

Conversion Context closure of the following

$$(\lambda x : A.M)N \longrightarrow_\beta M[x : A \mapsto N]$$

Typing rules

Context forming rules

$$\frac{}{-;- \text{ well-formed}}\text{Empty} \qquad \frac{\Sigma; - \vdash A : s \qquad c \notin \Sigma}{\Sigma, c : A; - \text{ well-formed}}\text{Decl} \qquad \frac{\Sigma; \Gamma \vdash A : s \qquad x \notin \Gamma}{\Sigma; \Gamma, x : A \text{ well-formed}}\text{Decl}$$

Conversion rule

$$\frac{\Sigma; \Gamma \vdash M : A \qquad \Sigma; \Gamma \vdash B : s \qquad A \equiv B}{\Sigma; \Gamma \vdash M : B}\text{Conv}$$

Regular type/term forming rules

$$\frac{\Sigma; \Gamma \text{ well-formed} \qquad (s_1, s_2) \in \mathcal{A}}{\Sigma; \Gamma \vdash s_1 : s_2}\text{Set}$$

$$\frac{\Sigma; \Gamma \vdash A : s_1 \qquad \Sigma; \Gamma, x : A \vdash B : s_2 \qquad (s_1, s_2, s_3) \in \mathcal{R}}{\Sigma; \Gamma \vdash \Pi x : A.B : s_3}\text{Prod}$$

$$\frac{\Sigma; \Gamma \text{ well-formed} \qquad x : A \in \Gamma}{\Sigma; \Gamma \vdash x : A}\text{Var}$$

$$\frac{\Sigma; \Gamma \text{ well-formed} \qquad c : A \in \Sigma}{\Sigma; \Gamma \vdash c : A}\text{Const}$$

$$\frac{\Sigma; \Gamma \vdash A : s_1 \qquad \Sigma; \Gamma, x : A \vdash B : s_2 \qquad \Sigma; \Gamma, x : A \vdash M : B \qquad (s_1, s_2, s_3) \in \mathcal{R}}{\Sigma; \Gamma \vdash \lambda x : A.M : \Pi x : A.B}\text{Abs}$$

$$\frac{\Sigma; \Gamma \vdash M : \Pi x : A.B \qquad \Sigma; \Gamma \vdash N : A}{\Sigma; \Gamma \vdash MN : B(N/x)}\text{App}$$

2 Administrative PTS

Terms

$$A, B, M, N ::= x \in \mathcal{X} \mid c \in \mathcal{C} \mid s \in \mathcal{S} \mid MN \mid \lambda x : A.M \mid \Pi x : A.B \mid M[x : A \mapsto N]$$

Conversion Context closure of the following

$$(\lambda x : A.M)N \longrightarrow_\beta M[x : A \mapsto N]$$

$$M[x : A \mapsto N] \longrightarrow_{subst} M(N/x)$$

Typing rules

Context forming rules

$$\frac{}{-;-\ \text{well-formed}}\ \text{Empty} \qquad \frac{\Sigma;-|-\vdash A : s \qquad c \notin \Sigma}{\Sigma, c : A;-\ \text{well-formed}}\ \text{Decl}$$

$$\frac{\Sigma;\Gamma|-\vdash A : s \qquad x \notin \Gamma}{\Sigma;\Gamma, x : A\ \text{well-formed}}\ \text{Decl}$$

Conversion rule

$$\frac{\Sigma;\Gamma|-\vdash M : A \qquad \Sigma;\Gamma|-\vdash B : s \qquad A \equiv B}{\Sigma;\Gamma|-\vdash M : B}\ \text{Conv}$$

Administrative rules

$$\frac{\Sigma;\Gamma, x : A|-\vdash M : B}{\Sigma;\Gamma|x : A \vdash M : B}\ \text{Struct}$$

$$\frac{\Sigma;\Gamma|x : A \vdash M : B \qquad \Sigma;\Gamma|-\vdash N : A}{\Sigma;\Gamma|-\vdash M[x : A \mapsto N] : B(N/x)}\ \text{Subst}$$

Regular type/term forming rules

$$\frac{\Sigma;\Gamma\ \text{well-formed} \qquad (s_1, s_2) \in \mathcal{A}}{\Sigma;\Gamma|-\vdash s_1 : s_2}\ \text{Set}$$

$$\frac{\Sigma;\Gamma|-\vdash A : s_1 \qquad \Sigma;\Gamma|x : A \vdash B : s_2 \qquad (s_1, s_2, s_3) \in \mathcal{R}}{\Sigma;\Gamma|-\vdash \Pi x : A.B : s_3}\ \text{Prod}$$

$$\frac{\Sigma;\Gamma\ \text{well-formed} \qquad x : A \in \Gamma}{\Sigma;\Gamma|-\vdash x : A}\ \text{Var}$$

$$\frac{\Sigma;\Gamma\ \text{well-formed} \qquad c : A \in \Sigma}{\Sigma;\Gamma|-\vdash c : A}\ \text{Const}$$

$$\frac{\Sigma;\Gamma|-\vdash A : s_1 \qquad \Sigma;\Gamma|x : A \vdash B : s_2 \qquad \Sigma;\Gamma|x : A \vdash M : B \qquad (s_1, s_2, s_3) \in \mathcal{R}}{\Sigma;\Gamma;|-\vdash \lambda x : A.M : \Pi x : A.B}\ \text{Abs}$$

$$\frac{\Sigma;\Gamma|-\vdash M : \Pi x : A.B \qquad \Sigma;\Gamma|-\vdash N : A}{\Sigma;\Gamma|-\vdash MN : B[x : A \mapsto N]}\ \text{App}$$

3 Soundness

Let $[\![-]\!] : \Lambda_{PTS} \to \Lambda_{APTS}$ be the injection of terms from a regular PTS to its corresponding administrative PTS.

Lemma 3.1 *Let $M, N \in \Lambda_{PTS}$. If $M \longrightarrow N$ then $[\![M]\!] \longrightarrow^* [\![N]\!]$.*

Proof. A beta step in PTS is simulated by a beta step in APTS followed by a subst step. ■

Corollary 3.1 *Let $M, N \in \Lambda_{PTS}$. If $M \equiv N$ then $[\![M]\!] \equiv [\![N]\!]$.*

We extend $[\![-]\!]$ to contexts, and then to judgements by defining

$$[\![\Sigma;\Gamma \vdash M : A]\!] := [\![\Sigma]\!];[\![\Gamma]\!]|-\vdash [\![M]\!] : [\![A]\!]\,.$$

Theorem 3.1 (Soundness) *Let Σ be a signature, Γ a contexts and M, A terms in a PTS. We have*

- *If $\Sigma; \Gamma$ well formed then $[\![\Sigma]\!]; [\![\Gamma]\!]$ well formed*
- *If $\Sigma; \Gamma \vdash M : A$ then $[\![\Sigma; \Gamma \vdash M : A]\!]$;*

Proof. The only case whose details are not totally clear is App. ■

4 Conservativity

We define the map $[\![-]\!] : \Lambda_{APTS} \to \Lambda_{PTS}$ by induction on M as

- $[\![M[x : A \mapsto N]]\!] := [\![M]\!]([\![N]\!]/x)$;
- $[\![MN]\!] := [\![M]\!][\![N]\!]$;
- etc...

We extend it to contexts and signatures, and then to judgements by the following definitions.

$$[\![\Sigma; \Gamma | - \vdash M : A]\!] = [\![\Sigma]\!]; [\![\Gamma]\!] \vdash [\![M]\!] : [\![A]\!]$$
$$[\![\Sigma; \Gamma | x : B \vdash M : A]\!] = [\![\Sigma]\!]; [\![\Gamma]\!], x : [\![B]\!] \vdash [\![M]\!] : [\![A]\!]$$

Lemma 4.1 *Let $M, N \in \Lambda_{APTS}$. If $M \longrightarrow N$ then $[\![M]\!] \longrightarrow [\![N]\!]$ or $[\![M]\!] = [\![N]\!]$.*

Corollary 4.1 *Let $M, N \in \Lambda_{APTS}$. If $M \equiv N$ then $[\![M]\!] \equiv [\![N]\!]$.*

Theorem 4.1 (Conservativity) *Let Σ be a signature, Γ a contexts and M, A terms in a APTS. We have*

- *If $\Sigma; \Gamma$ well formed then $[\![\Sigma]\!]; [\![\Gamma]\!]$ well formed*
- *If $\Sigma; \Gamma \vdash M : A$ then $[\![\Sigma; \Gamma \vdash M : A]\!]$;*

Proof. For the case Subst we apply the substitution lemma of the PTS. ■

5 Encoding APTS in Dedukti

$$\begin{aligned}
\text{(Sort-def)}\ & Sort : \textbf{Type} \\
\text{(U-def)}\ & U : Sort \to \textbf{Type} \\
\text{(El-def)}\ & El : (s : Sort) \to U s \to \textbf{Type} \\
\text{(Lmax-def)}\ & \sqcup :
\end{aligned}$$

Encoding

$$
\begin{aligned}
&[\![M]\!]_{\Sigma;\Gamma|x:A} = \lambda x : [A]_{\Sigma;\Gamma|-}.[\![M]\!]_{\Sigma;\Gamma,x:A|-}\\
&[\![c]\!]_{\Sigma;\Gamma|-} = c\\
&[\![x]\!]_{\Sigma;\Gamma|-} = x\\
&[\![s]\!]_{\Sigma;\Gamma|-} = |s|\\
&[\![\lambda x : A.M]\!]_{\Sigma;\Gamma|-} = abs\ |s_1|\ |s_2|\ |s_3|\ I\ [\![A]\!]_{\Sigma;\Gamma|-}\ [\![B]\!]_{\Sigma;\Gamma|x:A}\ [\![M]\!]_{\Sigma;\Gamma|x:A}\\
&[\![\Pi x : A.B]\!]_{\Sigma;\Gamma|-} = Prod\ |s_1|\ |s_2|\ |s_3|\ I\ [\![A]\!]_{\Sigma;\Gamma|-}\ [\![B]\!]_{\Sigma;\Gamma|x:A}\\
&[\![MN]\!]_{\Sigma;\Gamma|-} = app\ |s_1|\ |s_2|\ |s_3|\ I\ [\![A]\!]_{\Sigma;\Gamma|-}\ [\![B]\!]_{\Sigma;\Gamma|x:A}\ [\![M]\!]_{\Sigma;\Gamma|-}\ [\![N]\!]_{\Sigma;\Gamma|-}\\
&[\![M[x : A \mapsto N]]\!]_{\Sigma;\Gamma|-} = [\![M]\!]_{\Sigma;\Gamma|x:A}[\![N]\!]_{\Sigma;\Gamma|-}
\end{aligned}
$$

$$
\begin{aligned}
&[B]_{\Sigma;\Gamma|-} = El\ |s|\ [\![B]\!]_{\Sigma;\Gamma|-}\\
&[B]_{\Sigma;\Gamma|x:A} = \Pi x : [A]_{\Sigma;\Gamma|-}.[B]_{\Sigma;\Gamma,x:A|-}
\end{aligned}
$$

5.1 Soundness

Lemma 5.1 *Let $M, N \in \Lambda_{APTS}$. If $M \longrightarrow N$ then $[\![M]\!] \longrightarrow [\![N]\!]$.*

Corollary 5.1 *Let $M, N \in \Lambda_{APTS}$. If $M \equiv N$ then $[\![M]\!] \equiv [\![N]\!]$.*

Theorem 5.1 (Soundness) *Let Σ be a signature, Γ a contexts and M, A terms in an APTS. We have*

- *If $\Sigma; \Gamma$ well formed then $[\![\Sigma]\!]; [\![\Gamma]\!]$ well formed*
- *If $\Sigma; \Gamma|x : A \vdash M : B$ then $[\![\Sigma]\!]; [\![\Gamma]\!] \vdash [\![M]\!]_{\Sigma;\Gamma|x:A} : [B]_{\Sigma;\Gamma|x:A}$*
- *If $\Sigma; \Gamma|- \vdash M : B$ then $[\![\Sigma]\!]; [\![\Gamma]\!] \vdash [\![M]\!]_{\Sigma;\Gamma|-} : [B]_{\Sigma;\Gamma|-}$*

5.2 Some metatheory

Lemma 5.2 *If i is a level term with FV(i) $\subseteq \Theta$, then $-; \Theta; - \vdash i$ Lvl.*

Lemma 5.3 *If $\Sigma; \Theta; \Gamma, x : A \vdash M : B$ and $\Sigma; \Theta; \Gamma \vdash N : A$ then $\Sigma; \Theta; \Gamma \vdash M(N/x) : B$.*

Lemma 5.4 *If $\Sigma; \Theta, l; \Gamma \vdash M : B$ and $-; \Theta; - \vdash i$ Lvl then $\Sigma; \Theta; \Gamma \vdash M(i/l) : B$.*

Lemma 5.5 *If $\Sigma; \Theta; \Gamma \vdash$ Set$_i$ $: A$ and $i \longrightarrow_L j$ then $\Sigma; \Theta; \Gamma \vdash$ Set$_j$ $: A$.*

Proposition 5.1 *Let M and A be terms such that $\Sigma; \Theta; \Gamma \vdash M : A$. If $M \longrightarrow M'$ with a β, β_L or L step, then $\Sigma; \Theta; \Gamma \vdash M' : A$.*

Proposition 5.2 *$\longrightarrow_{\beta_L}$ is SN.*

6 A form of conservativity

It is clear that our universe polymorphic system extends the original one. More precisely, it is straightforward to verify that for any sequent

$$\Sigma; \Gamma \vdash M : A$$

that is provable on the original system, the sequent

$$\Sigma; -; \Gamma \vdash M : A$$

is provable on the polymorphic one.[1]

[1] More precisely, we must translate the occurrences of Set_n to $\mathsf{Set}_{S^n 0}$, bringing the universe levels from the metalanguage to the term language.

However, do we have some form of conservativity with respect to the original PTS? Obviously, we do not have regular conservativity, as polymorphic terms cannot even be expressed on the original system. Moreover, no term that depends on level variables cannot be expressed on the original system.

Therefore, no typing judgment $\Sigma; \Theta; \Gamma \vdash M : A$ where A is of the form $\forall l.A'$, or A or M contain a free level variable can be recovered. But what can we say about judgements not using neither of these features? More precisely, can every judgement $\Sigma; -; \Gamma \vdash M : A$ be translated back (up to interpreting the closed term levels as integers of the metalanguage)?

Unfortunately this is still insufficient. Indeed, even thought a term is not universe polymorphic, this does not mean that its construction does not use universe polymorphic terms. This can arise in two ways :

1. A polymorphic term can be introduced using the rule Gen, and then be instantiated at some concrete level to be used;

2. A polymorphic constant can be instantiated at some concrete level and also be used.

However, intuitively we can see that these features are not absolutely necessary when building regular non-polymorphic terms. Indeed, if we introduce a polymorphic term using Gen to then instantiate it using Inst, this correspond in some sense to a cut, and thus we can try to find a procedure to eliminate them, as we do in natural deduction proofs.

The use of a polymorphic constant is also unnecessary if we instead replace it by a constant for each universe. Of course, we cannot have infinite signatures, but as in a proof we can only use finitly many constants we can just take a large enough signature to contain all of the constants we use.

The fact that universe polymorphism allows us to replace an infinity of declarations by just one is actually one of the main reasons we wanted to have it. But of course, as we have just discussed, universe polymorphism is not a necessity, as instead of declaring a universe polymorphic identity, we can just declare one identity function for each level we need. Getting rid of universe polymorphism should make things more difficult, but intuitively we see that there is no reason this would not be possible.

Indeed, we will show that this is actually possible by getting rid of both points (1.) and (2.) when building proofs of closed-level terms which are not of the form $\forall l.A$.

Eliminating UP redexes We define a universe polymorphic redex (UP redex) as a term of the form $(\Lambda l.M)i$. Differently from beta redexes, UP redexes are very easy to eliminate, as each β_L reduction step cannot create new UP redexes.

n occurrence of the rule Gen followed by an occurrence of the rule Inst, as shown bellow.

$$\cfrac{\cfrac{\Sigma; \Theta; \Gamma \vdash \forall l.A : \mathsf{Set}_\omega \qquad \Sigma; \Theta, l; \Gamma \vdash M : A}{\Sigma; \Theta; \Gamma \vdash \Lambda l.M : \forall l.A}\,\mathsf{Gen} \qquad -; \Theta; - \vdash i \ \mathsf{Lvl}}{\Sigma; \Theta; \Gamma \vdash Mi : A(i/l)}\,\mathsf{Inst}$$

We claim that if we can get rid of UP cuts, then we can get rid of all occurrences of the Gen rule. Indeed, as we are supposing that A is not of the form $\forall l.A'$, the derivation cannot end with an occurrence of Gen, and thus there are three possible cases: Gen is followed by Gen, Inst or Conv.

Lemma 6.1 *Let M, A be terms with no level variables and A not of the form $\forall l.A'$. If $\pi : \Sigma, -, \Gamma \vdash M : A$ contains an occurrence of Gen, we can find a derivation π' in which this rule is followed by Inst.*

Proof. As we are supposing that A is not of the form $\forall l.A'$, the derivation cannot end with an occurrence of Gen, and thus there are three possible cases: Gen is followed by Gen, Inst or Conv.

Now consider an occurrence of Gen which is not followed by any other one Gen. If it is followed by any number of occurrences of Conv, we can merge all these occurrences to obtain a derivation of the form

$$\cfrac{\cfrac{\Sigma; -; \Gamma \vdash \forall l.A : \mathsf{Set}_\omega \qquad \Sigma; l; \Gamma \vdash M : A}{\Sigma; -; \Gamma \vdash \Lambda l.M : \forall l.A}\,\mathsf{Gen} \qquad \Sigma; -; \Gamma \vdash B : \mathsf{Set}_\alpha \qquad A \equiv B}{\Sigma; -; \Gamma \vdash M : B}\,\mathsf{Conv}$$

in which the rule Conv is not followed neither by Conv or Gen. However, if $A \equiv B$, then it is clear that B must be of the form $\forall l.C$ with $A \equiv C$. Therefore, the only possible rule we can use is Inst, which gives a derivation of the following form.

$$\cfrac{\cfrac{\cfrac{\Sigma; -; \Gamma \vdash \forall l.A : \mathsf{Set}_\omega \qquad \Sigma; l; \Gamma \vdash M : A}{\Sigma; -; \Gamma \vdash \Lambda l.M : \forall l.A}\,\mathsf{Gen} \qquad \Sigma; -; \Gamma \vdash \forall l.C : \mathsf{Set}_\omega \qquad \forall l.A \equiv \forall l.C}{\Sigma; -; \Gamma \vdash M : \forall l.C}\,\mathsf{Conv} \qquad -; -; - \vdash i\ \mathsf{Lvl}}{\Sigma; -; \Gamma \vdash Mi : C(i/l)}\,\mathsf{Ins}$$

Now, note that because $A \equiv C$, we have that $A(i/l) \equiv C(i/l)$. Moreover, as $\Sigma; \Theta; \Gamma \vdash \forall l.C : \mathsf{Set}_\omega$, we also have $\Sigma; \Theta, l; \Gamma \vdash C : \mathsf{Set}_\alpha$ for some α. By applying x, we conclude that we can derive $\Sigma; \Theta; \Gamma \vdash C(i/l) : \mathsf{Set}_\alpha$.

Therefore, we can replace this part of the derivation by the following one, yielding an occurrence of Gen followed by Inst.

$$\cfrac{\cfrac{\cfrac{\Sigma; -; \Gamma \vdash \forall l.A : \mathsf{Set}_\omega \qquad \Sigma; l; \Gamma \vdash M : A}{\Sigma; -; \Gamma \vdash \Lambda l.M : \forall l.A}\,\mathsf{Gen} \qquad -; -; - \vdash i\ \mathsf{Lvl}}{\Sigma; -; \Gamma \vdash Mi : A(i/l)}\,\mathsf{Inst} \qquad \Sigma; -; \Gamma \vdash C(i/l) : \mathsf{Set}_\alpha \qquad A(i/l) \equiv C(i/l)}{\Sigma; -; \Gamma \vdash M : C(i/l)}$$

■

First note that, because M, A do not contain free level variables, the level terms appearing in them are generated by $0, \mathsf{S}$ and $\sqcup$. Therefore, we can interpret the levels terms appearing on them by the integers of the metalanguage of the original system.

We are closer to some form of conservativity, but this is still not enough. In particular, there are still two problems that must be treated.

Dealing with universe polymorphic variables and constants

Nevertheless, because universe quantification always appear in prenex form, we cannot do much with universe polymorphic terms other then applying them to universe levels. This observation shows that, even thought polymorphism is not part of the language of the original system, if we use

7 Encoding on the $\lambda\Pi$-calculus

In order to express the UP system on the $\lambda\Pi$-calculus, we will build our encoding little by little.

First, before trying to express the types of UP, we need to express sorts, which type the types. For this we declare a type of sorts and a function that embeds them on the world of types, so we can use the to type the types of UP.

$$\begin{aligned}&\mathsf{Sort} : \textbf{Type}\\&\mathsf{U} : \mathsf{Sort} \to \textbf{Type}\end{aligned}$$

Now we can declare the sorts of UP as being elements of Sort. However, to do that we need a way of encoding the levels on which the sort Set depends. For this we declare a type of levels and all of the possible ways we can construct a level term.

$$\begin{aligned}&\mathsf{L} : \textbf{Type}\\&0 : \mathsf{L}\\&\mathsf{S} : \mathsf{L} \to \mathsf{L}\\&\sqcup : \mathsf{L} \to \mathsf{L} \to \mathsf{L}\end{aligned}$$

Note that we don't need to add a constructor for variables, because we can just use the variables of the $\lambda\Pi$-calculus of type L as level variables. We also do not declare yet any rewriting rule associated to levels, as this is a rather complicated question which we leave for another section.

Now that we have a type for levels we can declare our sorts. We represent the sorts Set_i as functions taking a term of type L and producing a sort. We also add a constant to represent the sort Set_ω.

$$\begin{aligned}&\mathsf{Set} : \mathsf{L} \to \mathsf{Sort}\\&\mathsf{Set}_\omega : \mathsf{Sort}\end{aligned}$$

Now we can type types using this sorts. For instance, a type A of sort Set_0 should be declared as an element $[\![A]\!] : \mathsf{U}\ (\mathsf{Set}\ 0)$.

Now, if M is an element of the type A we cannot declare its translation $[\![M]\!]$ as being typed by $[\![A]\!]$ because $[\![A]\!]$ is not a type. To do this on the $\lambda\Pi$-calculus we need to transform $[\![A]\!]$ into a type. For this, we declare the function

$$\mathsf{El} : (s : \mathsf{Sort}) \to \mathsf{U}\ s \to \textbf{Type}$$

which embeds the inhabitants of each $\mathsf{U}\ s$ into **Type**. Now we can declare $[\![M]\!]$ as being an element of $\mathsf{El}\ (\mathsf{Set}\ 0)\ [\![A]\!]$.

We now have a base structure for dealing with the sorts Set_i, but we still need to explain how they are related to each other. More precisely, we need to express that a sort of the form Set_i should inhabit $\mathsf{Set}_{\mathsf{S}i}$. To do this we would like to state $\mathsf{Set}\ i : \mathsf{U}\ (\mathsf{Set}\ (\mathsf{S}\ i))$, but his is not possible because $\mathsf{Set}\ i$ has already the type Sort.

To solve this we should then have two copies of each sort. For this, we declare the family of constants

$$\mathsf{set} : (i : \mathsf{L}) \to \mathsf{U}\ (\mathsf{Set}\ (\mathsf{S}\ i))$$

and the rule

$$\mathsf{El}\ \_\ (\mathsf{set}\ i) \longrightarrow \mathsf{U}\ (\mathsf{Set}\ i)$$

stating that the element set i of U (Set (S i)) is mapped to the type U (Set i) by El. This encodes the relationship between each set i and each Set i.

We that our representation of the sorts Set_i is more structured, we can try to add an encoding for products, that is, encoding the following rule.

$$\frac{\Sigma; \Theta; \Gamma \vdash A : \mathsf{Set}_i \qquad \Sigma; \Theta; \Gamma, x : A \vdash B : \mathsf{Set}_j}{\Sigma; \Theta; \Gamma \vdash \Pi x : A.B : \mathsf{Set}_{i \sqcup j}}\ \mathsf{Prod}$$

For this, we add a constant

$$\Pi : (i\ j : \mathsf{L}) \to (A : \mathsf{U}\ (\mathsf{Set}\ i)) \to (\mathsf{El}\ (\mathsf{Set}\ i)\ A \to \mathsf{U}\ (\mathsf{Set}\ j)) \to (\mathsf{U}\ (\mathsf{Set}\ (i \sqcup j)))$$

which, given a type A : U (Set i)) and a type U (Set j)) which depends on an element of type El (Set i) A, allows us to build a new type that lives in (U (Set ($i \sqcup j$))).

In order to express that the elements of this type really are product types, we also add the following rule.

$$\Pi\ i\ j\ A\ B \longrightarrow \Pi a : (\mathsf{El}\ i\ A).\mathsf{El}\ j\ (B\ a)$$

Finally, we also need to declare the rules of universe polymorphism. For this, we need to encode the rule Omega, which allows us to abstract a type with a free level variable into a universe polymorphic type.

$$\frac{\Sigma; \Theta, l; \Gamma \vdash A : \mathsf{Set}_\alpha}{\Sigma; \Theta; \Gamma \vdash \forall l.A : \mathsf{Set}_\omega}\ \mathsf{Omega}$$

To do this we add the constant

$$\forall : (\alpha : \mathsf{L} \to \mathsf{Sort}) \to ((i : \mathsf{L}) \to \mathsf{U}\ (\alpha\ i)) \to \mathsf{U}\ \mathsf{Set}_\omega$$

and the rule

$$\mathsf{El}\ \_\ (\forall\ \alpha\ A) \longrightarrow (i : \mathsf{L}) \to \mathsf{El}\ (\alpha\ i)\ (A\ i)\,.$$

7.1 Defining the encoding

We resume the signature and the reduction rules of the translation by the following list.

Encoding of levels

(L-def) $\mathsf{L} : \textbf{Type}$

(L0-def) $0 : \mathsf{L}$

(LS-def) $\mathsf{S} : \mathsf{L} \to \mathsf{L}$

(Lmax-def) $\sqcup : \mathsf{L} \to \mathsf{L} \to \mathsf{L}$

Encoding of sorts

(Sort-def) $\mathsf{Sort} : \textbf{Type}$

(Set-def) $\mathsf{Set} : \mathsf{L} \to \mathsf{Sort}$

(Omega-def) $\mathsf{Set}_\omega : \mathsf{Sort}$

$\mathsf{U} : \mathsf{Sort} \to \textbf{Type}$

$\mathsf{El} : (s : \mathsf{Sort}) \to \mathsf{U}\ s \to \textbf{Type}$

$\mathsf{set} : (i : \mathsf{L}) \to \mathsf{U}\ (\mathsf{Set}\ (\mathsf{S}\ i))$

$\mathsf{El}\ \_\ (\mathsf{set}\ i) \longrightarrow \mathsf{U}\ (\mathsf{Set}\ i)$

Encoding of products

$\Pi : (i\ j : \mathsf{L}) \to (A : \mathsf{U}\ (\mathsf{Set}\ i)) \to (\mathsf{El}\ (\mathsf{Set}\ i)\ A \to \mathsf{U}\ (\mathsf{Set}\ j)) \to (\mathsf{U}\ (\mathsf{Set}\ (i \sqcup j)))$

$\Pi\ i\ j\ A\ B \longrightarrow \Pi a : (\mathsf{El}\ i\ A).\mathsf{El}\ j\ (B\ a)$

Encoding of universe polymorphism

$\forall : (\alpha : \mathsf{L} \to \mathsf{Sort}) \to ((i : \mathsf{L}) \to \mathsf{U}\ (\alpha\ i)) \to \mathsf{U}\ \mathsf{Set}_\omega$

$\mathsf{El}\ \_\ (\forall\ \alpha\ A) \longrightarrow (i : \mathsf{L}) \to \mathsf{El}\ (\alpha\ i)\ (A\ i)$

The goal is to define a translation so that we can translate well typed terms. Therefore, we cannot just define a function $\Lambda_{\mathsf{UP}} \to \Lambda_{\lambda\Pi/\equiv}$, as we this wouldn't be a total function. A second try would be to define a function $\Lambda_{\mathsf{UP}}^{\text{well-typed}} \to \Lambda_{\lambda\Pi/\equiv}$, which is only defined for well typed terms. However, as we will see, in order to define the translation of a type M we need both to have a type which it inhabits and a sort for this type.

The correct definition will be to define a function which takes three terms M, A, s such that

Definition 7.1 (Encoding of terms of UP) *We define a term translation function*

$$[\![-]\!] : \Lambda_{UP}^{\textit{well-typed}} \to \Lambda_{\lambda\Pi/\equiv}$$

for the well types terms of UP inductively by the following.

$$\begin{aligned}
[\![x]\!] &= x\\
[\![c]\!] &= c\\
[\![M\ N]\!] &= [\![M]\!][\![N]\!]\\
[\![\lambda x : A.M]\!] &= \lambda x : \mathit{El}\ s_A\ [\![A]\!].[\![M]\!]\\
[\![\Pi x : A.B]\!] &= \Pi\ s_A\ s_B(x : \mathit{El}\ s_A\ [\![A]\!]) \to \mathit{El}\ s_B\ [\![B]\!]\\
[\![\mathit{Set}_i]\!] &= \mathit{set}\ [i]\\
[\![\Lambda l.M]\!] &= \lambda l : \mathit{L}.[\![M]\!]\\
[\![\forall l]\!] &= (l : \mathit{L}) \to [\![M]\!]
\end{aligned}$$

Now that we have a type for levels we can declare our sorts. We represent the sorts Set_i as functions taking a term of type L and producing a sort. We also add a constant to represent the sort Set_ω.

8 Non-Prenex UP

8.1 Metatheory LambdaPiM

Proposition 8.1 (Weakening) *If $\Sigma; \Gamma \vdash M : A$ and $\Sigma; \Gamma, \Gamma'$ well-formed then $\Sigma; \Gamma, \Gamma' \vdash M : A$.*

Theorem 8.1 *If $\Sigma; \Gamma \vdash A' : s$ and $A \equiv_R A'$ then*

- *$\Sigma; \Gamma, x : A, \Gamma' \vdash M : B$ implies $\Sigma; \Gamma, x : A', \Gamma' \vdash M : B$*
- *$\Sigma; \Gamma, x : A, \Gamma'$ well-formed implies $\Sigma; \Gamma, x : A', \Gamma'$ well-formed*

Proof. By induction on the derivation tree. For all cases it is trivial, except with var and decl.

- Var : If $M \neq x$, it suffices to apply the IH. If $M = x$, then we have

$$\frac{\Sigma; \Gamma, x : A, \Gamma' \text{ well-formed} \qquad x : A \in \Gamma, x : A, \Gamma'}{\Sigma; \Gamma, x : A, \Gamma' \vdash x : A}\,\text{Var}$$

  By IH, $\Sigma; \Gamma, x : A', \Gamma'$ well-formed, and thus

$$\frac{\Sigma; \Gamma, x : A', \Gamma' \text{ well-formed} \qquad x : A' \in \Gamma, x : A', \Gamma'}{\Sigma; \Gamma, x : A', \Gamma' \vdash x : A'}\,\text{Var}$$

  As $\Sigma; \Gamma \vdash A' : s$ and $\Sigma; \Gamma, x : A', \Gamma'$ well-formed, by weakening we have $\Sigma; \Gamma, x : A', \Gamma' \vdash A' : s$ and thus by conv we conclude

$$\frac{\Sigma; \Gamma, x : A', \Gamma' \vdash x : A' \qquad \Sigma; \Gamma, x : A', \Gamma' \vdash A' : s \qquad A \equiv_R A'}{\Sigma; \Gamma, x : A', \Gamma' \vdash x : A}\,\text{Conv}$$

- Decl : If Γ' is not empty, it suffices to apply the IH. If $\Gamma' = -$, then we have

$$\frac{\Sigma; \Gamma \text{ well-formed} \qquad \Sigma; \Gamma \vdash x : A}{\Sigma; \Gamma, x : A \text{ well-formed}}\,\text{Decl}$$

  By hypothesis we have $\Sigma; \Gamma \vdash x : A'$, and as $\Sigma; \Gamma$ well-formed, we conclude with Decl

$$\frac{\Sigma; \Gamma \text{ well-formed} \qquad \Sigma; \Gamma \vdash x : A'}{\Sigma; \Gamma, x : A' \text{ well-formed}}\,\text{Decl}$$

■

Theorem 8.2 *If $\Sigma \vdash A' : s$, $A \equiv_R A'$ and $\Sigma, c : A, \Sigma'; \Gamma \vdash M : B$ then we have*

$$\Sigma, c : A, \Sigma'; \Gamma \vdash M : B\,.$$

Proof. Similar to Thm 5.2 ■

8.2 System

Levels

$$i, j ::= 0 \mid \mathsf{S}\ i \mid i \sqcup j \mid l \in \mathcal{L}\,,$$

Levels and Omega

$$s ::= \mathsf{Set}_i \mid \mathsf{Prop}_i \mid \mathsf{Set}_{\omega_n}$$

Terms

$$A, B, M, N ::= x \in \mathcal{X} \mid c \in \mathcal{C} \mid s \mid MN \mid \lambda x : A.M \mid \Pi x : A.B \mid M \cdot i \mid \Lambda l.M \mid \forall l.A$$

Conversion

$$\begin{aligned}(\lambda x : A.M)N &\longrightarrow_\beta M(N/x)\\ (\Lambda l.M)\cdot i &\longrightarrow_{\beta_L} M(l/i)\\ i \equiv_L j &\iff \forall \sigma, [\![i]\!]_\sigma = [\![j]\!]_\sigma\end{aligned}$$

Before presenting the rules we define the function $\square$, which gives the successor of a given sort

$$\begin{aligned}\square\mathsf{Set}_i &= \mathsf{Set}_{Si}\\ \square\mathsf{Prop}_i &= \mathsf{Prop}_{Si}\\ \square\mathsf{Set}_{\omega_n} &= \mathsf{Set}_{\omega_{n+1}}\end{aligned}$$

and we define the function $\vee$, which gives the product of two sorts

$$\begin{aligned}k_i \vee k'_j &= k'_{i\sqcup j} && \text{with } k, k' \in \{\mathsf{Set}, \mathsf{Prop}\}\\ k_i \vee \mathsf{Set}_{\omega_n} &= \mathsf{Set}_{\omega_n} && \text{with } k \in \{\mathsf{Set}, \mathsf{Prop}\}\\ \mathsf{Set}_{\omega_n} \vee k_i &= \mathsf{Set}_{\omega_n} && \text{with } k \in \{\mathsf{Set}, \mathsf{Prop}\}\\ \mathsf{Set}_{\omega_m} \vee \mathsf{Set}_{\omega_n} &= \mathsf{Set}_p && \text{with } p = \max\{n, m\}\end{aligned}$$

Note that this two functions are just aliases, as on the definition we have equality and not conversion. The symbols $\diamond, \square$ are not even part of the grammar, they are just here to allow us to speak about successors and products in a more homogeneous way.

Signature forming rules

$$\frac{}{-;\Theta;- \text{ well-formed}}\ \text{Empty} \qquad \frac{\Sigma;\Theta;-\vdash c : A \qquad c \notin \Sigma}{\Sigma, c : A;\Theta;- \text{ well-formed}}\ \text{Decl}$$

Context forming rules

$$\frac{}{\Sigma;\Theta;- \text{ well-formed}}\ \text{Empty} \qquad \frac{\Sigma;\Theta;\Gamma\vdash A : s \qquad x \notin \Gamma}{\Sigma;\Theta;\Gamma, x : A \text{ well-formed}}\ \text{Decl}$$

Conversion rule[2]

$$\frac{\Sigma;\Theta;\Gamma\vdash M : A \qquad \Sigma;\Theta;\Gamma\vdash A : s \qquad \Sigma;\Theta;\Gamma\vdash B : s \qquad A \equiv B}{\Sigma;\Theta;\Gamma\vdash M : B}\ \text{Conv}$$

[2]By showing confluence, subject reduction and strong normalization for this system, we can show that the second hypothesis is admissible

Level forming rules

$$\frac{l \in \Theta}{-;\Theta;-\vdash l \ \mathsf{Lvl}}\ \text{Lvar} \qquad \frac{}{-;\Theta;-\vdash 0 \ \mathsf{Lvl}}\ \text{L0}$$

$$\frac{-;\Theta;-\vdash i \ \mathsf{Lvl}}{-;\Theta;-\vdash \mathsf{S}\ i \ \mathsf{Lvl}}\ \text{LS} \qquad \frac{-;\Theta;-\vdash i \ \mathsf{Lvl} \qquad -;\Theta;-\vdash j \ \mathsf{Lvl}}{-;\Theta;-\vdash i \sqcup j \ \mathsf{Lvl}}\ \text{Lmax}$$

Regular type/term forming rules

$$k = \mathsf{Set}, \mathsf{Prop}\ \frac{-;\Theta;-\vdash i \ \mathsf{Lvl} \qquad \Sigma;\Theta;\Gamma \text{ well-formed}}{\Sigma;\Theta;\Gamma\vdash k_i : k_{\mathsf{S}i}}\ \text{Sort}$$

$$\frac{\Sigma;\Theta;\Gamma\vdash A : s \qquad \Sigma;\Theta;\Gamma, x : A\vdash B : s'}{\Sigma;\Theta;\Gamma\vdash \Pi x : A.B : s \vee s'}\ \text{Prod}$$

$$\frac{\Sigma;\Theta;\Gamma \text{ well-formed} \qquad x : A \in \Gamma}{\Sigma;\Theta;\Gamma\vdash x : A}\ \text{Var}$$

$$\frac{\Sigma;\Theta;\Gamma\ \text{well-formed} \qquad c : A \in \Sigma}{\Sigma;\Theta;\Gamma \vdash c : A}\ \textsf{Const}$$

$$\frac{\Sigma;\Theta;\Gamma \vdash A : s \qquad \Sigma;\Theta;\Gamma, x : A \vdash B : s' \qquad \Sigma;\Theta;\Gamma, x : A \vdash M : B}{\Sigma;\Theta;\Gamma \vdash \lambda x : A.M : \Pi x : A.B}\ \textsf{Abs}$$

$$\frac{\Sigma;\Theta;\Gamma \vdash M : \Pi x : A.B \qquad \Sigma;\Theta;\Gamma \vdash N : A}{\Sigma;\Theta;\Gamma \vdash MN : B(N/x)}\ \textsf{App}$$

Universe polymorphism rules

$$\frac{\Sigma;\Theta;\Gamma\ \text{well-formed}}{\Sigma;\Theta;\Gamma \vdash \textsf{Set}_{\omega_n} : \textsf{Set}_{\omega_{n+1}}}\ \textsf{SortOmega}$$

$$k = \textsf{Set}, \textsf{Prop}\ \frac{\Sigma;\Theta, l;\Gamma \vdash A : k_i \qquad \Sigma;\Theta;\Gamma\ \text{well-formed}}{\Sigma;\Theta;\Gamma \vdash \forall l.A : \textsf{Set}_{\omega}}\ \textsf{Omega1}$$

$$\frac{\Sigma;\Theta, l;\Gamma \vdash A : \textsf{Set}_{\omega_n} \qquad \Sigma;\Theta;\Gamma\ \text{well-formed}}{\Sigma;\Theta;\Gamma \vdash \forall l.A : \textsf{Set}_{\omega_n}}\ \textsf{Omega2}$$

$$\frac{\Sigma;\Theta;\Gamma\ \text{well-formed} \qquad \Sigma;\Theta, l;\Gamma \vdash M : A}{\Sigma;\Theta;\Gamma \vdash \Lambda l.M : \forall l.A}\ \textsf{Gen}$$

$$\frac{\Sigma;\Theta;\Gamma \vdash M : \forall l.A \qquad -;\Theta;- \vdash i\ \textsf{Lvl}}{\Sigma;\Theta;\Gamma \vdash M \cdot i : A(i/l)}\ \textsf{Inst}$$

8.3 Metatheory

Proposition 8.2 (Inversion) *Let $\Sigma;\Theta;\Gamma \vdash M : A$.*

- *If $M = x$ then for some A' with $A \equiv A'$ we have $x : A' \in \Gamma$*
- *If $M = c$ then for some A' with $A \equiv A'$ we have $c : A' \in \Sigma$*
- *If $M = s$ then $A \equiv_L \square\ s$*
- *If $M = N\ P$ then there are B, C, x with $A \equiv C(P/x)$ and*
  - $\Sigma;\Theta;\Gamma \vdash N : \Pi x : B.C$
  - $\Sigma;\Theta;\Gamma \vdash P : B$
- *If $M = \lambda x : B.N$ then for some C with $A \equiv \Pi x : B.C$ we have*
  - $\Sigma;\Theta;\Gamma \vdash B : s$
  - $\Sigma;\Theta;\Gamma, x : B \vdash C : s'$
  - $\Sigma;\Theta;\Gamma, x : B \vdash N : C$
- *If $M = \Pi x : A.B$ then for some s, s' we have $A \equiv_L s \vee s'$ with*
  - $\Sigma;\Theta;\Gamma \vdash A : s$
  - $\Sigma;\Theta;\Gamma, x : A \vdash B : s'$
- *If $M = \Lambda l.M$ then there is B with $A \equiv \forall l.B$ and*
  - $\Sigma;\Theta, l;\Gamma \vdash M : B$
- *If $M = \forall l.B$, either there is i, A with $A = Set_\omega$ and*
  - $\Gamma;\Theta, l;\Sigma \vdash B : k_i$ *with $k = Set$ or $Prop$*

or $A = \mathit{Set}_{\omega_n}$ for some $n \in \mathbb{N}$ and

- $\Gamma; \Theta, l; \Sigma \vdash B : \mathit{Set}_{\omega_n}$

Proof. By induction on the derivation tree. ■

Proposition 8.3 *If $-; \Theta; - \vdash i$ Lvl and $i \equiv_L j$ then $-; \Theta; - \vdash j$ Lvl*

Proof. If suffices to verify that if $i \equiv_L j$ then the same variables appear in them. ■

Corollary 8.1 *If $\Sigma; \Theta; \Gamma \vdash M : A$, $M \equiv_L M'$ and $A \equiv_L A'$ then $\Sigma; \Theta; \Gamma \vdash M' : A'$.*

Proposition 8.4 *If $\Sigma; \Theta; \Gamma \vdash M : A, A'$ then $A \equiv A'$.*

Proof. By induction on the first derivation tree.

- SortOmega, Omega1, Omega2, Conv : Trivial
- Var, Const, Sort : By inversion on the second derivation tree
- Prod : A consequence of
- Abs :

■

Corollary 8.2 *If $\Sigma; \Theta; \Gamma \vdash A : s, s'$ then $s \equiv_L s'$.*

Proof. Follows from the fact that, if s, s' are sorts, the only conversion that can be applied is the level conversion. ■

9 Encoding

Encoding of levels

(L-def)	L : **Type**
(L0-def)	0 : L
(LS-def)	S : L $\to$ L
(Lmax-def)	$\sqcup$: L $\to$ L $\to$ L

Encoding of meta-naturals

(N-def)	N : **Type**	
(Nzero-def)	zero : N	
(Nsucc-def)	succ : N $\to$ N	
(Nmax-def)	max : N $\to$ N $\to$ N	
(Nmax-rule1)	max zero (succ x)	$\longrightarrow x$
(Nmax-rule2)	max (succ x) zero	$\longrightarrow x$
(Nmax-rule3)	max (succ x) (succ y)	$\longrightarrow$ suc (max x y)

Encoding of sorts

(Sort-def) Sort : **Type**

(Set-def) Set : L $\to$ Sort

(Prop-def) Prop : L $\to$ Sort

(SetOmega-def) Set_ω : N $\to$ Sort

(U-def) U : Sort $\to$ **Type**

(El-def) El : $(s$: Sort$)$ $\to$ U $s \to$ **Type**

Encoding of sort rule

(Rule-def) $\vee$: Sort $\to$ Sort $\to$ Sort

(Rule-rule1) (Set i) $\vee$ (Set j) $\longrightarrow$ Set $(i \sqcup j)$

(Rule-rule1) (Prop i) $\vee$ (Set j) $\longrightarrow$ Set $(i \sqcup j)$

(Rule-rule1) (Prop i) $\vee$ (Prop j) $\longrightarrow$ Prop $(i \sqcup j)$

(Rule-rule1) (Set i) $\vee$ (Prop j) $\longrightarrow$ Prop $(i \sqcup j)$

(Rule-rule1) (Set_ω n) $\vee$ (Set_ω m) $\longrightarrow$ Set_ω (max n m)

(Rule-rule1) (Set_ω n) $\vee$ (Prop _) $\longrightarrow$ Set_ω n

(Rule-rule1) (Set_ω n) $\vee$ (Set _) $\longrightarrow$ Set_ω n

(Rule-rule1) (Prop _) $\vee$ (Set_ω n) $\longrightarrow$ Set_ω n

(Rule-rule1) (Set _) $\vee$ (Set_ω n) $\longrightarrow$ Set_ω n

Encoding of products

$\Pi : (s\ s'$: Sort$) \to (A$: U $s) \to ($El $s\ A \to$ U $s') \to ($U $(s \vee s'))$

$\Pi\ s\ s'\ A\ B \longrightarrow \Pi a : ($El $s\ A).$El $s'\ (B\ a)$

Encoding of universe polymorphism

$\forall_S : (\alpha : \text{L} \to \text{L}) \to ((i : \text{L}) \to \text{U (Set } (\alpha\ i))) \to \text{U Set}_\omega$

$\forall_P : (\alpha : \text{L} \to \text{L}) \to ((i : \text{L}) \to \text{U (Prop } (\alpha\ i))) \to \text{U Set}_\omega$

$\forall_\omega : (n : \text{N}) \to (\text{L} \to (\text{U } (\text{Set}_\omega\ n))) \to \text{U } (\text{Set}_\omega\ n)$

El _ $(\forall\ \alpha\ A) \longrightarrow (i : \text{L}) \to$ El $(\alpha\ i)\ (A\ i)$

We write Σ_{UP} for the signature of the encoding in $\lambda\Pi/R$.

We define the encoding of a judgement $\Sigma; \Theta; \Gamma \vdash M : A$ by induction on M.

$$
\begin{array}{rll}
[\![x]\!]_{\Sigma;\Theta;\Gamma} \ni x & & \\
[\![c]\!]_{\Sigma;\Theta;\Gamma} \ni c & & \\
[\![s]\!]_{\Sigma;\Theta;\Gamma} \ni \hat{\diamond}\ [\![s]\!] & & \\
[\![i]\!]_{\Sigma;\Theta;\Gamma} \ni [\![i]\!] & & \\
[\![MN]\!]_{\Sigma;\Theta;\Gamma} \ni [M]_{\Sigma;\Theta;\Gamma}\ [N]_{\Sigma;\Theta;\Gamma} & & \\
[\![\lambda x : A.M]\!]_{\Sigma;\Theta;\Gamma} \ni \lambda x : \mathsf{El}\ [\![s_A]\!]\ [A]_{\Sigma;\Theta;\Gamma}.[M]_{\Sigma;\Theta;\Gamma,x:A} & & \text{if } \Sigma;\Theta;\Gamma \vdash A : s_A \\
[\![\Pi x : A.B]\!]_{\Sigma;\Theta;\Gamma} \ni \hat{\Pi}\ [\![s_A]\!]\ [\![s_B]\!]\ [A]_{\Sigma;\Theta;\Gamma}\ (\lambda x : \mathsf{El}\ [\![s_A]\!]\ [A]_{\Sigma;\Theta;\Gamma}.[B]_{\Sigma;\Theta;\Gamma,x:A}) & & \text{if } \Sigma;\Theta;\Gamma \vdash A : s_A, B : s_B \\
[\![\Lambda l.M]\!]_{\Sigma;\Theta;\Gamma} \ni \lambda l : \mathsf{L}.[M]_{\Sigma;\Theta,l;\Gamma} & & \\
[\![M \cdot i]\!]_{\Sigma;\Theta;\Gamma} \ni [M]_{\Sigma;\Theta;\Gamma}\ [i] & & \\
[\![\forall l.A]\!]_{\Sigma;\Theta;\Gamma} \ni \hat{\forall}_S\ (\lambda l : \mathsf{L}.[\![i]\!])\ (\lambda l : \mathsf{L}.[\![A]\!]_{\Sigma;\Theta,l;\Gamma}) & & \text{if } \Sigma;\Theta,l;\Gamma \vdash A : \mathsf{Set}_i \\
[\![\forall l.A]\!]_{\Sigma;\Theta;\Gamma} \ni \hat{\forall}_P\ (\lambda l : \mathsf{L}.[\![i]\!])\ (\lambda l : \mathsf{L}.[A]_{\Sigma;\Theta,l;\Gamma}) & & \text{if } \Sigma;\Theta,l;\Gamma \vdash A : \mathsf{Prop}_i \\
[\![\forall l.A]\!]_{\Sigma;\Theta;\Gamma} \ni \hat{\forall}_\omega\ [\![n]\!]\ (\lambda l : \mathsf{L}.[\![A]\!]_{\Sigma;\Theta,l;\Gamma}) & & \text{if } \Sigma;\Theta,l;\Gamma \vdash A : \mathsf{Set}_{\omega_n}
\end{array}
$$

$$
\begin{array}{rl}
\text{Signature} & \\
[\![\Sigma = c_1 : A_1, ..., c_k : A_k]\!] \ni \Sigma_{UP}, c_1 : \mathsf{El}\ [s_1]\ [A_1]_{\Sigma;-;-}, ..., c_k : \mathsf{El}\ [s_k]\ [A_k]_{\Sigma;-;-} & \text{if } \Sigma;-;- \vdash A_i : s_i \\
\text{Level vars} & \\
[\![\Sigma;-]\!] \ni [\Sigma]; - & \\
[\![\Sigma;\Theta', l]\!] \ni [\Sigma;\Theta'], l : \mathsf{L} & \\
\text{Context} & \\
[\![\Sigma;\Theta;-]\!] \ni [\Sigma;\Theta] & \\
[\![\Sigma;\Theta;\Gamma', x : A]\!] \ni [\Sigma;\Theta;\Gamma'], x : \mathsf{El}\ s_A\ A & \text{if } \Sigma;\Theta;\Gamma' \vdash A : s_A
\end{array}
$$

Note that the signature Σ is translated as a signature, whereas the level vars Θ and the context Γ are translated in the context. Indeed, the character ';', which separates the signature from the context, is introduced in $[\![\Sigma;-]\!]$. That is, informally the translation of $\Sigma;\Theta;\Gamma$ is $\Sigma_{UP}, [\Sigma]; [\Theta], [\Gamma]$.

Assumption 9.1 *If $i \equiv_L j$ then $[i] \equiv_L [j]$.*

Theorem 9.1 *If $N_1, N_2 \in [\![M]\!]_{\Sigma;\Theta;\Gamma}$ then $N_1 \equiv_L N_2$.*

Proof. By induction in M. We do the case $\Pi x : A.B$.

We have that the N_i are of the form

$$\hat{\Pi}\ [s_A^{1,i}]\ [s_B^i]\ A^{1,i}\ (\lambda x : \mathsf{El}\ [s_A^{2,i}]\ A^{2,i}.B^i)\,.$$

where

- $A^{1,1}, A^{1,2}, A^{2,1}, A^{2,2} \in [\![A]\!]_{\Sigma;\Theta;\Gamma,x:A}$
- $B^1, B^2 \in [\![B]\!]_{\Sigma;\Theta;\Gamma}$
- for $s = s_A^{1,1}, s_A^{1,2}, s_A^{2,1}, s_A^{2,2}$ we have $\Sigma;\Theta;\Gamma \vdash A : s$

- for $s = s_B^1, s_B^2$ we have $\Sigma; \Theta; \Gamma \vdash B : s$,

By IH, $A^{1,1} \equiv_L A^{1,2}, A^{2,1} \equiv_L A^{2,2}$ and $B^1 \equiv_L B^2$. By Corollary 5.1, $s_A^{1,1} \equiv_L s_A^{1,2}, s_A^{2,1} \equiv_L s_A^{2,2}$ and $s_B^1 \equiv_L s_B^2$. Therefore, we deduce $N_1 \equiv_L N_2$.

■

9.1 Correctness of the translation

Theorem 9.2 *If $-; \Theta; - \vdash i$ Lvl then for all $(\Sigma_{UP}; \hat{\Theta}) \in [\![-; \Theta; -]\!]$ we have*

$$\Sigma_{UP}; \hat{\Theta} \vdash [\![i]\!] : \textsf{L}\,.$$

Theorem 9.3 *Let $-; \Theta; - \vdash i$ Lvl. If $i \equiv_L j$ then for all $(\Sigma_{UP}; \hat{\Theta}) \in [\![-; \Theta; -]\!]$ we have*

$$\Sigma_{UP}; \hat{\Theta} \vdash [\![j]\!] : \textsf{L}\,.$$

Proof. A consequence of the fact that $\equiv_L$ preserves the free variables. ■

Theorem 9.4 *If $\Sigma_{UP}, \hat{\Sigma}; \hat{\Theta}, \hat{\Gamma} \vdash \textsf{El}\ [\![s]\!]\ \hat{A} : \boldsymbol{s}$ and $\textsf{El}\ [\![s]\!]\ \hat{A} \equiv_L \textsf{El}\ [\![s']\!]\ \overline{A}$ then*

$$\Sigma_{UP}, \hat{\Sigma}; \hat{\Theta}, \hat{\Gamma} \vdash \textsf{El}\ [\![s']\!]\ \overline{A} : \boldsymbol{s}\,.$$

Corollary 9.1 *Let $\Sigma; \Theta; \Gamma \vdash M : A$. Let $(\Sigma_{UP}, \hat{\Sigma}; \hat{\Theta}, \hat{\Gamma}) \in [\![\Sigma; \Theta; \Gamma]\!]$ and*

Theorem 9.5 *Let $\Sigma; \Theta; \Gamma$ be a context.*

1. *If $\Sigma; \Theta; \Gamma$ well-formed then for $(\Sigma_{UP}, \hat{\Sigma}; \hat{\Theta}, \hat{\Gamma}) \in [\![\Sigma; \Theta; \Gamma]\!]$ we have*

$$\Sigma_{UP}, \hat{\Sigma}; \hat{\Theta}, \hat{\Gamma} \text{ well-formed}\,.$$

2. *If $\Sigma; \Theta; \Gamma \vdash M : A$ and $\Sigma; \Theta; \Gamma \vdash A : s$ then for $(\Sigma_{UP}, \hat{\Sigma}; \hat{\Theta}, \hat{\Gamma}) \in [\![\Sigma; \Theta; \Gamma]\!]$, $\hat{M} \in [\![M]\!]_{\Sigma;\Theta;\Gamma}$ and $\hat{A} \in [\![A]\!]_{\Sigma;\Theta;\Gamma}$*

$$\Sigma_{UP}, \hat{\Sigma}; \hat{\Theta}, \hat{\Gamma} \vdash \hat{M} : \textsf{El}\ [\![s]\!]\ \hat{A}\,.$$

Proof. By induction on the derivation.

Var : We have

$$\frac{\Sigma; \Theta; \Gamma \text{ well-formed} \qquad x : A \in \Gamma}{\Sigma; \Theta; \Gamma \vdash x : A}\,\text{Var}$$

Let $(\Sigma_{UP}, \hat{\Sigma}; \hat{\Theta}, \hat{\Gamma}) \in [\![\Sigma; \Theta; \Gamma]\!]$ and $\hat{A} \in [\![A]\!]_{\Sigma;\Theta;\Gamma}$. Let s be a sort with $\Sigma; \Theta; \Gamma \vdash A : s$. We need to show $\Sigma_{UP}, \hat{\Sigma}; \hat{\Theta}, \hat{\Gamma} \vdash x : \textsf{El}\ [\![s]\!]\ \hat{A}$.

We have $\hat{\Gamma} = \hat{\Gamma_1}, x : \textsf{El}\ [\![s']\!]\ \overline{A}, \hat{\Gamma_2}$, where by definition of the translation, $\overline{A} \in [\![A]\!]_{\Sigma;\Theta;\Gamma_1}$ and $\Sigma; \Theta; \Gamma_1 \vdash \overline{A} : s'$.

By IH, $\Sigma_{UP}, \hat{\Sigma}; \hat{\Theta}, \hat{\Gamma}$ well-formed. Thus, by rule Var, we have

$$\frac{\Sigma_{UP}, \hat{\Sigma}; \hat{\Theta}, \hat{\Gamma} \text{ well-formed} \qquad x : \textsf{El}\ [\![s']\!]\ \overline{A} \in \Gamma}{\Sigma_{UP}, \hat{\Sigma}; \hat{\Theta}, \hat{\Gamma} \vdash x : \textsf{El}\ [\![s']\!]\ \overline{A}}\,\text{Var}$$

By x, y we have $[\![s]\!] \equiv_L [\![s']\!]$ and $\hat{A} \equiv_L \overline{A}$, and thus, by th z,

$$\Sigma_{UP}, \hat{\Sigma}; \hat{\Theta}, \hat{\Gamma} \vdash x : \mathsf{El}\ [\![s]\!]\ \hat{A}\,.$$

Const :

$$\frac{\Sigma; \Theta; \Gamma \text{ well-formed} \qquad c : A \in \Sigma \qquad \Sigma; -; - \vdash A : s}{\Sigma; \Theta; \Gamma \vdash c : A}\ \mathsf{Const}$$

■