Perspectives on Martin-Löf Type Theory and Univalent Type Theory

Iosif Petrakis

University of Bern
Winter term 17/18

Contents

CHAPTER 1

Identity in Martin-Löf's intensional type theory

1.1. The J-rule

DEFINITION 1.1.1. Form$_{x=_A y}$: If $x : A$ and $y : A$, the **equality type** $x =_A y : \mathcal{U}$.
Intro$_{x=_A x}$:

$$\mathtt{refl}_A : \prod_{x:A} x =_A x.$$

Ind$_{=_A}$: If

$$C : \prod_{x,y:A} \prod_{p:x=_A y} \mathcal{U}$$

is a dependent family of types in $\mathcal{U}$, and if

$$c : \prod_{x:A} C(x, x, \mathtt{refl}_x)$$

is a dependent function, there is a dependent function

$$F : \prod_{x,y:A} \prod_{p:x=_A y} C(x, y, p)$$

such that

$$F(x, x, \mathtt{refl}_x) \equiv c(x).$$

This is the inductive definition of the type family $=_A : A \to A \to \mathcal{U}$ with two indices in A and with constructor

$$\frac{x : A}{\mathtt{refl}_x : x =_A x}$$

The type $x =_A y$ is *not* defined inductively, but the type family is.

In the rest A, B denote always some types in the universe $\mathcal{U}$.

DEFINITION 1.1.2. We call the following judgement and its associated computation rule

$$J : \prod_{C:\prod_{x,y:A}\prod_{p:x=_A y}\mathcal{U}} \prod_{c:\prod_{x:A} C(x,x,\mathtt{refl}_x)} \prod_{x,y:A} \prod_{p:x=_A y} C(x, y, p)$$

$$J(C, c, x, x, \mathtt{refl}_x) \equiv c(x), \quad x : A$$

the J-judgement and the J-computation rule.

1.2. Some fundamental implications of the J-rule

DEFINITION 1.2.1. The judgement $a : A$ *implies* the judgement $b : B$, if from $a : A$ we can derive a judgement $b' : B$. In this case, if r is a computation rule of $a : A$ and s a computation rule of $b : B$, we say that r *implies* s, if from r we can derive s', where s' is same to s, but with respect to b'. The judgements are *equivalent*, if one implies the other, while the computations rules of two equivalent judgements are *equivalent*, if one rule implies the other.

PROPOSITION 1.2.2. *The J-judgement and the J-computation rule imply the following* `LeastRefl`*-judgement and* `LeastRefl`*-computation rule, respectively,*

$$\mathtt{LeastRefl} : \prod_{R:A\to A\to\mathcal{U}} \prod_{r:\prod_{x:A} R(x,x)} \prod_{x,y:A} \prod_{p:x=_A y} R(x,y),$$

$$\mathtt{LeastRefl}(R, r, x, x, \mathtt{refl}_x) \equiv r(x), \quad x : A.$$

PROOF. □

As Coquand mentions in [**6**], the J-judgement is just the `LeastRefl`-judgement together with the presence of proof-terms.

PROPOSITION 1.2.3. *The J-judgement and the J-computation rule imply the following* `Transport`*-judgement and* `Transport`*-computation rule, respectively,*

$$\mathtt{Transport} : \prod_{P:A\to\mathcal{U}} \prod_{x,y:A} \prod_{p:x=_A y} (P(x) \to P(y))$$

$$\mathtt{Transport}(P, x, x, \mathtt{refl}_x) \equiv \mathrm{id}_{P(x)}, \quad x : A.$$

PROOF. □

If A and x, y are fixed we use the notation p^P_* for $\mathtt{Transport}(P, x, y, p)$.

PROPOSITION 1.2.4. *The* `LeastRefl` *and the* `Transport`*-judgement are equivalent. Moreover, the* `LeastRefl`*-computation rule is equivalent to the* `Transport`*-computation rule.*

PROOF. □

PROPOSITION 1.2.5. *Let the dependent functions* $C : \prod_{x,y:A} \prod_{p:x=_A y} \mathcal{U}$ *and* $c : \prod_{x:A} C(x, x, \mathtt{refl}_x)$. *If* $F, G : \prod_{x,y:A} \prod_{p:x=_A y} C(x, y, p)$ *such that*

$$F(x, x, \mathtt{refl}_x) \equiv c(x) \equiv G(x, x, \mathtt{refl}_x),$$

the following judgement and computation rule are derived

$$H : \prod_{x,y:A} \prod_{p:x=_A y} F(x, y, p) =_{C(x,y,p)} G(x, y, p),$$

$$H(x, x, \mathtt{refl}_x) \equiv \mathtt{refl}_{c(x)}, \quad x : A.$$

PROOF. □

In the proof of Proposition 1.2.4 we have e.g., that from the `LeastRefl`-judgement we get a judgement $T : \prod_{P:A\to\mathcal{U}} \prod_{x,y:A} \prod_{p:x=_A y} (P(x) \to P(y))$, and this term T is not necessarily judgementaly identical to `Transport`, a fact that explains our formulation of Definition 1.2.1. Through Proposition 1.2.5 though T is pointwise equal to `Transport`, and by function extensionality it is propositionally equal to it.

PROPOSITION 1.2.6. *The J-judgement and the J-computation rule imply the following* `Application`*-judgement and* `Application`*-computation rule, respectively,*

$$\mathtt{Application} : \prod_{f:A\to B} \prod_{x,y:A} \prod_{p:x=_A y} f(x) =_B f(y)$$

$$\mathtt{Application}(f, x, x, \mathtt{refl}_x) \equiv \mathtt{refl}_{f(x)}, \quad x : A.$$

PROOF. □

The standard notation for $\mathtt{Application}(f)$ is $\mathtt{ap}_f$.

1.3. The based J-rule

DEFINITION 1.3.1. We call the following judgement and its rule

$$j : \prod_{a:A} \prod_{C:\prod_{x:A} \prod_{p:a=_A x} \mathcal{U}} \prod_{c:C(a,\mathtt{refl}_a)} \prod_{x:A} \prod_{p:a=_A x} C(x, p)$$

$$j(a, C, c, a, \mathtt{refl}_a) \equiv c$$

the j-judgement and the j-computation rule. Usually, we denote $j(a)$ by j_a, for every $a : A$.

For the following judgements and their corresponding computation rules

$$\texttt{leastrefl} : \prod_{a:A} \prod_{R_a:A\to\mathcal{U}} \prod_{r_a:R_a(a)} \prod_{x:A} \prod_{p:a=_A x} R_a(x)$$

$$\texttt{leastrefl}(a, R_a, r_a, a, \texttt{refl}_a) \equiv r_a,$$

$$\texttt{transport} : \prod_{a:A} \prod_{P:A\to\mathcal{U}} \prod_{x:A} \prod_{p:a=_A x} (P(a) \to P(x))$$

$$\texttt{transport}(a, P, a, \texttt{refl}_a) \equiv \mathrm{id}_{P(a)},$$

one can prove the corresponding Propositions 1.2.2−1.2.4. We use the notations $\texttt{leastrefl}_a \equiv \texttt{leastrefl}(a)$ and $\texttt{transport}_a \equiv \texttt{transport}(a)$, for every $a : A$.

Definition 1.3.2. A family of judgments $(i_a : P(a))_{a:A}$ *implies* the judgement $b : B$, if given $(i_a : P(a))_{a:A}$, we can derive a judgement $b' : B$. In this case, if r_a is a computation rule of $i_a : P(a)$ and s is a computation rule adjusted to $b : B$, the family of computation rules $r(a)_{a:A}$ *implies* s, if from $r(a)_{a:A}$ we can derive s', where s' is same to s, but with respect to b'. The converse implications are defined similarly.

Proposition 1.3.3. *The family of judgements*

$$\left(j_a : \prod_{C:\prod_{x:A}\prod_{p:a=_A x}\mathcal{U}} \prod_{c:C(a,\texttt{refl}_a)} \prod_{x:A} \prod_{p:a=_A x} C(x,p)\right)_{a:A}$$

implies the judgement

$$J : \prod_{C:\prod_{x,y:A}\prod_{p:x=_A y}\mathcal{U}} \prod_{c:\prod_{x:A} C(x,x,\texttt{refl}_x)} \prod_{x,y:A}\prod_{p:x=_A y} C(x,y,p),$$

and the family of computation rules $(j_a(C, c, a, \texttt{refl}_a) \equiv c)_{a:A}$ *implies the computation rule* $J(C, c, x, x, \texttt{refl}_x) \equiv c(x)$, *for every* $x : A$.

Proof. □

Theorem 1.3.4. *The converse of Proposition 1.3.3 holds.*

Proof. (Altenkirch, Coguen)

□

1.4. Dependent pair types ($\sum$-types)

DEFINITION 1.4.1. Form$_{\sum}$: Given $A : \mathcal{U}$ and a family $P : A \to \mathcal{U}$, the dependent pair type $\sum_{x:A} P(x)$ is in $\mathcal{U}$.

Intro$_{\sum}$:

$$\frac{x : A,\ u : P(x)}{(x, a) : \sum_{x:A} P(x)}$$

Rec$_{\sum}$: If $C : \mathcal{U}$ and $G : \prod_{x:A}(P(x) \to C)$, there exists

$$F : \left(\sum_{x:A} P(x)\right) \to C$$

such that

$$F((x, u)) :\equiv G(x, u),$$

for every $x : A$ and $u : P(x)$.

Ind$_{\sum}$: If $Q : (\sum_{x:A} P(x)) \to \mathcal{U}$, and

$$G : \prod_{x:A} \prod_{u:P(x)} Q((x, u)),$$

there is a dependent function

$$F : \prod_{u:\sum_{x:A} P(x)} Q(u)$$

such that

$$F((x, u)) :\equiv G(x, u),$$

for every $x : A$ and $u : P(x)$.

Next the existence of the two projection functions of a $\sum$-type is shown.

PROPOSITION 1.4.2. *The judgements* $\mathtt{pr}_1 : (\sum_{x:A} P(x)) \to A$ *and* $\mathtt{pr}_2 : \prod_{u:\sum_{x:A} P(x)} P(\mathtt{pr}_1(u))$ *with the following corresponding computation rules are derivable*

$$\mathtt{pr}_1((x, u)) :\equiv x, \quad x : A, u : P(x),$$

$$\mathtt{pr}_2((x, u)) :\equiv u, \quad x : A, u : P(x).$$

PROOF. With the use of Rec$_{\sum}$, if $G : \prod_{x:A}(P(x) \to A)$ is defined by

$$G(x, u) :\equiv x,$$

for every $x : A$, the first projection function

$$\mathtt{pr}_1 : \left(\sum_{x:A} P(x)\right) \to A$$

is defined by

$$\mathtt{pr}_1((x,u)) :\equiv G(x,u) \equiv x,$$

for every $x : A$ and $u : P(x)$. With the use of $\mathrm{Ind}_{\sum}$, if $Q : (\sum_{x:A} P(x)) \to \mathcal{U}$ is defined by

$$Q(u) :\equiv P(\mathtt{pr}_1(u)),$$

for every $u : \sum_{x:A} P(x)$, and if $G : \prod_{x:A} \prod_{u:P(x)} P(\mathtt{pr}_1(u))$ is defined by

$$G(x,u) :\equiv u,$$

for every $x : A$ and $u : P(x)$, then the second projection function

$$\mathtt{pr}_2 : \prod_{u:\sum_{x:A} P(x)} P(\mathtt{pr}_1(u))$$

is defined by

$$\mathtt{pr}_2((x,u)) :\equiv G(x,u) \equiv u,$$

for every $x : A$ and $u : P(x)$. □

PROPOSITION 1.4.3. *The following judgement and computation rule are derivable*

$$M : \prod_{z:\sum_{x:A} P(x)} \left(z =_{\sum_{x:A} P(x)} (\mathtt{pr}_1(z), \mathtt{pr}_2(z))\right)$$
$$M\big((x,u)\big) \equiv \mathtt{refl}_{(x,u)}, \quad x : A, u : P(x).$$

PROOF. □

1.5. The M-rule and the based M-rule

PROPOSITION 1.5.1. *If The J-judgement and the J-computation rule imply the following M-judgement and M-computation rule, respectively,*

$$M : \prod_{a,x:A} \prod_{p:a=_A x} (a, \mathtt{refl}_a) =_{E_a} (x,p)$$
$$M(a,a,\mathtt{refl}_a) \equiv \mathtt{refl}_{(a,\mathtt{refl}_a)},$$

where

$$E_a \equiv \sum_{x:A} (a =_A x).$$

PROOF. □

Similarly we get that the j-judgement and the j-computation rule imply the following m-judgement and m-computation rule, respectively,

$$m : \prod_{a:A} \prod_{u:E_a} (a, \mathtt{refl}_a) =_{E_a} u$$

$$m_a\Big((a, \mathtt{refl}_a)\Big) \equiv \mathtt{refl}_{(a,\mathtt{refl}_a)},$$

where $m_a \equiv m(a)$.

PROPOSITION 1.5.2. *The family of judgements*

$$\left(m_a : \prod_{u:E_a} (a, \mathtt{refl}_a) =_{E_a} u\right)_{a:A}$$

implies the judgement

$$M : \prod_{a,x:A} \prod_{p:a=_A x} (a, \mathtt{refl}_a) =_{E_a} (x, p),$$

and the family of computation rules $\big(m_a((a, \mathtt{refl}_a)) \equiv \mathtt{refl}_{(a,\mathtt{refl}_a)}\big)_{a:A}$ *implies the computation rule* $M(a, a, \mathtt{refl}_a) \equiv \mathtt{refl}_{(a,\mathtt{refl}_a)}$, *for every* $a : A$.

PROOF. We define $M(a, x, p) \equiv m_a((x, p))$. □

THEOREM 1.5.3. *The converse of Proposition 1.5.2 holds.*

PROOF. If we define the family $Q : E_a \to \mathcal{U}$ by $Q(u) \equiv (a, \mathtt{refl}_a) =_{E_a} u$, for every $u : E_a$, then

$$M(a) : \prod_{x:A} \prod_{p:a=_A x} \big((a, \mathtt{refl}_a) =_{E_a} (x, p)\big) \equiv \prod_{x:A} \prod_{p:a=_A x} Q((x, p)).$$

By $\mathrm{Ind}_{\sum}$ there is $F : \prod_{u:E_a} Q(u)$ such that

$$F((a, \mathtt{refl}_a)) \equiv M(a)(a, \mathtt{refl}_a) \equiv M(a, a, \mathtt{refl}_a) \equiv \mathtt{refl}_{(a,\mathtt{refl}_a)}.$$

□

THEOREM 1.5.4. *The following two judgements*

$$m_a : \prod_{u:E_a} (a, \mathtt{refl}_a) =_{E_a} u$$

$$\mathtt{transport}_a : \prod_{P:A\to\mathcal{U}} \prod_{x:A} \prod_{p:a=_A x} (P(a) \to P(x))$$

imply the judgement

$$j_a : \prod_{C:\prod_{x:A}\prod_{p:a=_A x}\mathcal{U}} \prod_{c:C(a,\mathtt{refl}_a)} \prod_{x:A} \prod_{p:a=_A x} C(x,p)$$

and the same holds for their corresponding computation rules.

Proof. Let the judgements $C : \prod_{x:A}\prod_{p:a=_A x}\mathcal{U}$ and $c : C(a,\mathtt{refl}_a)$. We find $F : \prod_{x:A}\prod_{p:a=_A x} C(x,p)$ such that $F(a,\mathtt{refl}_a) \equiv c$. Let $P : E_a \to \mathcal{U}$ defined by $P((x,p)) \equiv C(x,p)$, for every $x : A$ and $p : a =_A x$. The existence of P follows from $\mathrm{Rec}_{\sum}$. Since

$$m_a((x,p)) : (a,\mathtt{refl}_a) =_{E_a} (x,p)$$

using $\mathtt{transport}_a$ we have that

$$\big[m_a((x,p))\big]^P_* : P((a,\mathtt{refl}_a)) \to P((x,p)) \equiv C(a,\mathtt{refl}_a) \to C(x,p),$$

and we define

$$F \equiv \lambda(x : A, p : a =_A x).\big[m_a((x,p))\big]^P_*(c).$$

Then we get

$$\begin{aligned} F(a,\mathtt{refl}_a) &\equiv \big[m_a((a,\mathtt{refl}_a))\big]^P_*(c) \\ &\equiv \big[\mathtt{refl}_{(a,\mathtt{refl}_a)}\big]^P_*(c) \\ &\equiv \mathrm{id}_{P((a,\mathtt{refl}_a)}(c) \\ &\equiv \mathrm{id}_{C(a,\mathtt{refl}_a)}(c) \\ &\equiv c. \end{aligned}$$

□

As a corollary of Theorem 1.5.4 we get an immediate proof of Theorem 1.5.3 as follows. Let $a : A$. From the J-judgement we get the M-judgement (Proposition 1.5.1) and from the M-judgement we get the m_a-judgement (Theorem 1.5.3). From the J-judgement we get the `Transport`-judgement (Proposition 1.2.3) and from it we get easily the $\mathtt{transport}_a$-judgement. Then we get the j_a-judgement from Theorem 1.5.4.

Corollary 1.5.5 (Coquand, 2014). *The following judgements and corresponding computation rules are equivalent:*
(i) J.
(ii) `Transport` *and* M.
(iii) `LeastRefl` *and* M.

PROOF. For the equivalence between (i) and (ii) it suffices to show that `Transport` and M imply J. Since `Transport` and M imply $\texttt{transport}_a$ and m_a, for every $a : A$, respectively, by Theorem 1.5.4, we get j_a, for every $a : A$, hence by Proposition 1.3.3 we get J. The equivalence between (ii) and (iii) is Proposition 1.2.4. □

Of course, a similar equivalence holds between the judgements and computation rules j_a, $\texttt{transport}_a$ and m_a, $\texttt{leastrefl}_a$ and m_a.

1.6. The J-judgement and the Yoneda lemma

In [**14**] Rijke viewed a type family $P : A \to \mathcal{U}$ over $A : \mathcal{U}$ as a presheave of a locally small category $\mathcal{C}$ i.e., as an element of $\mathbf{Set}^{\mathcal{C}^{\mathrm{op}}}$, and he gave a type-theoretic version of the Yoneda lemma. Recall that $\mathbf{Set}^{\mathcal{C}^{\mathrm{op}}}$ denotes the category of contravariant set-valued functors on $\mathcal{C}$, and by the definition of a locally small category $\mathcal{C}$, if $A, B \in \mathcal{C}_0$, where $\mathcal{C}_0$ denotes the objects of $\mathcal{C}$ and $\mathcal{C}_1$ denotes the arrows of $\mathcal{C}$, the collection

$$\mathrm{Hom}_{\mathcal{C}}(A, B) \equiv \{f \in \mathcal{C}_1 \mid f : A \to B\}$$

is a set. Note of course, that the universe of types is closed under exponentiation. According to the Yoneda lemma (see [**3**], section 8.2), if $\mathcal{C}$ is a locally small category, $C \in \mathcal{C}_0$ and $F \in \mathbf{Set}^{\mathcal{C}^{\mathrm{op}}}$, then there is an isomorphism

$$\mathrm{Hom}_{\mathbf{Set}^{\mathcal{C}^{\mathrm{op}}}}(\mathcal{Y}(C), F) \simeq F(C),$$

which is natural in both F and C, where

$$\mathcal{Y} : \mathcal{C} \to \mathbf{Set}^{\mathcal{C}^{\mathrm{op}}}$$

is the Yoneda embedding i.e., the functor

$$\mathcal{Y}(C) \equiv \mathrm{Hom}_{\mathcal{C}}(-, C) : \mathcal{C}^{\mathrm{op}} \to \mathbf{Set}$$

$$\mathcal{Y}(f : C \to C') \equiv \mathrm{Hom}_{\mathcal{C}}(-, f) : \mathrm{Hom}_{\mathcal{C}}(-, C) \to \mathrm{Hom}_{\mathcal{C}}(-, C')$$

defined post-compositionally. Through the Yoneda lemma the Yoneda embedding is shown to be an embedding i.e., an injective on objects, faithful, and full functor.

DEFINITION 1.6.1. If $P, Q : A \to \mathcal{U}$ are type families on A and $a : A$, we define

$$\mathrm{Hom}(P, Q) \equiv \prod_{x:A} \big(P(x) \to Q(x)\big)$$

$$\mathcal{Y}_a : A \to \mathcal{U}$$

$$\mathcal{Y}(a)(x) \equiv x =_A a, \quad x : A.$$

Hence, we have that

$$\begin{aligned}\mathrm{Hom}(\mathcal{Y}(a), P) &\equiv \prod_{x:A} \big(\mathcal{Y}(a) \to P(x)\big) \\ &\equiv \prod_{x:A} \big((x =_A a) \to P(x)\big) \\ &\equiv \prod_{x:A} \prod_{p:x=_A a} P(x).\end{aligned}$$

Proposition 1.6.2.

Theorem 1.6.3 (Yoneda lemma in ITT + Function extensionality (Rijke, 2012)). *Let $P : A \to \mathcal{U}$ and $a : A$. There is a pair of quasi-inverses*

$$(j, i) : \mathrm{Hom}(\mathcal{Y}(a), P) \simeq P(a)$$

i.e.,

$$\begin{aligned}(j \circ i)(u) &= u, \quad u : P(a), \\ (i \circ j)(\sigma) &= \sigma, \quad \sigma : \prod_{x:A} \prod_{p:x=_A a} P(x)\end{aligned}$$

such that

$$\begin{aligned}i(u)(a, \mathtt{refl}_a) &\equiv u, \quad u : P(a), \\ j(\sigma) &\equiv \sigma(a, \mathtt{refl}_a), \quad \sigma : \prod_{x:A} \prod_{p:x=_A a} P(x).\end{aligned}$$

Proof. □

Definition 1.6.4. We call the judgement $(j, i) : \mathrm{Hom}(\mathcal{Y}(a), P) \simeq P(a)$ the *$\mathcal{Y}$-judgement* and the families of computation rules

$$\big(i(u)(a, \mathtt{refl}_a) \equiv u\big)_{u:P(a)},$$

$$\big(j(\sigma) \equiv \sigma(a, \mathtt{refl}_a)\big)_{\sigma:\prod_{x:A} \prod_{p:x=_A a} P(x)}$$

the *left* and *right $\mathcal{Y}$-computation rule*, respectively.

Proposition 1.6.5. *The $\mathcal{Y}$-judgement implies the introduction rule of the equality type i.e., the inhabitedness of the type $a =_A a$, for every $a : A$.*

Proof. If $a : A$, and if we consider as P in the type-theoretic Yoneda lemma the type family $\mathcal{Y}$, then

$$\mathrm{Hom}(\mathcal{Y}(a), \mathcal{Y}(a)) \equiv \left(\prod_{x:A} \prod_{p:x=_A a} x =_A a\right) \simeq (a =_A a) \equiv \mathcal{Y}(a).$$

The only element of $\mathrm{Hom}(\mathcal{Y}(a), \mathcal{Y}(a))$ we can determine at this point is

$$R \equiv \lambda(x : A, p : x =_A a).p$$

and $j(R) : a =_A a$. □

We can view $j(R)$ as the $\mathtt{refl}_a$. Note that if we use the right $\mathcal{Y}$-computation rule, which has meaning after Proposition 1.6.5, we get that $j(R) \equiv R(a, \mathtt{refl}_a) \equiv \mathtt{refl}_a$.

PROPOSITION 1.6.6. *The $\mathcal{Y}$-judgement implies the* `Transport`*-judgement and the left $\mathcal{Y}$-computation rule implies the* `Transport`*-rule.*

PROOF. □

LEMMA 1.6.7. *If $B : \mathcal{U}$, the $\mathcal{Y}$-judgement and the $\mathcal{Y}$-computation rules imply the following judgement and corresponding computation rules:*

$$(j, i) : \left(\prod_{x:A} \prod_{p:x=_A a} B\right) \simeq B$$

$$i(b)(a, \mathtt{refl}_a) \equiv b, \quad b : B,$$

$$j(\sigma) \equiv \sigma(a, \mathtt{refl}_a), \quad \sigma : \prod_{x:A} \prod_{p:x=_A a} B.$$

Moreover, if $b : B$, $x : A$, and $p : x =_A a$, then

$$i(b)(x, p) =_B b.$$

PROOF. □

COROLLARY 1.6.8. *The $\mathcal{Y}$-judgement with the $\mathcal{Y}$-computation rules imply the M-judgement.*

PROOF. From Lemma 1.6.7 we have that

$$\left(\prod_{x:A} \prod_{p:x=_A a} {E_a}'\right) \simeq {E_a}',$$

$${E_a}' \equiv \sum_{x:A} x =_A a.$$

Let $\sigma : \prod_{x:A} \prod_{p:x=_A a} {E_a}'$ be defined as

$$\sigma \equiv \lambda(x : A, p : x =_A a).(x, p).$$

Since

$$\sigma = i(j(\sigma)) \equiv i(\sigma(a, \mathtt{refl}_a)),$$

by Lemma 1.6.7 we get

$$(x, p) \equiv \sigma(x, p) =_{E_a'} [i(\sigma(a, \mathtt{refl}_a))](x, p) =_{E_a'} \sigma(a, \mathtt{refl}_a) \equiv (a, \mathtt{refl}_a).$$

□

The next theorem is shown without the use of function extensionality.

THEOREM 1.6.9 (Escardó, 2015). *The J-judgement follows from the $\mathcal{Y}$-judgement and the $\mathcal{Y}$-computation rules.*

PROOF. It follows immediately from Proposition 1.6.6, Corollary 1.6.8, and Corollary 1.5.5. □

CHECK the M-computation rule, it depends on the formulation of function extensionality, check Voevodsky's, or standard version, and the Yoneda version of extensionality.

1.7. Notes

CHAPTER 2

The axiom of univalence

2.1. Voevodsky's axiom of univalence

DEFINITION 2.1.1. If $P : A \to \mathcal{U}$ and $F, G : \prod_{x:A} P(x)$ a *homotopy* H between F, G is a term of type

$$H : F \sim G \equiv \prod_{x:A} \big(F(x) =_{P(x)} G(x)\big).$$

In the special case of a constant type family, if $f, g : A \to B$ a homotopy H between f, g is a term of type

$$H : f \sim g \equiv \prod_{x:A} \big(f(x) =_B g(x)\big).$$

PROPOSITION 2.1.2. *If* $A, B : \mathcal{U}, f, g : A \to B, H : f \sim g$, *and* $p : x =_A y$, *then*

$$H(x) * \mathtt{ap}_g(p) = \mathtt{ap}_f(p) * H(y).$$

Hence, if A, B are seen as categories, f, g are functors and a homotopy $H : f \sim g$ is a natural transformation between f, g.

DEFINITION 2.1.3. If $A, B : \mathcal{U}$ and $f : A \to B$, we define "f is a *quasi-inverse*" as the type

$$\mathrm{qinv}(f) \equiv \sum_{g:B\to A} [(f \circ g \sim \mathrm{id}_B) \times (g \circ f \sim \mathrm{id}_A)],$$

and "f is an *equivalence*", or "f is a *Voevodsky equivalence*", as the type

$$\mathtt{isequiv}(f) \equiv \left(\sum_{g:B\to A} (f \circ g) \sim \mathrm{id}_B)\right) \times \left(\sum_{h:B\to A} (h \circ f) \sim \mathrm{id}_A)\right).$$

The *weak equivalence* between A, B is the type

$$A \simeq_{\mathrm{w}} B \equiv \sum_{f:A\to B} \mathrm{qinv}(f),$$

and the *equivalence*, or the *Voevodsky-equivalence*, between A, B is the type

$$A \simeq_{\mathcal{U}} B \equiv \sum_{f:A\to B} \mathtt{isequiv}(f).$$

Clearly, if

$$e_A \equiv \big((\mathrm{id}_A, H), (\mathrm{id}_A, H)\big)$$
$$H \equiv \lambda(x:A).\mathtt{refl}_x,$$

then

$$e_A : \mathtt{isequiv}(\mathrm{id}_A),$$

hence $A \simeq_{\mathcal{U}} A$. If e is a canonical element of $A \simeq_{\mathcal{U}} B$, we write

$$e \equiv (e^*, e^{**})$$
$$e^* : A \to B, \quad e^{**} : \mathtt{isequiv}(e^*).$$

PROPOSITION 2.1.4. *Let $A, B : \mathcal{U}$ and $f : A \to B$.*
(i) $\mathrm{qinv}(f) \leftrightarrow \mathtt{isequiv}(f)$.
(ii) $\prod_{e_1,e_2:\mathtt{isequiv}(f)}(e_1 = e_2)$.

Because of Proposition 2.1.4 one can use a simpler writing for the terms of type $A \simeq_{\mathcal{U}} B$ using only the function terms. It is easy to show that if $f : A \simeq_{\mathcal{U}} B$, then $f^{-1} : B \simeq_{\mathcal{U}} A$, and if $g : B \simeq_{\mathcal{U}} C$, then $g \circ f : A \simeq_{\mathcal{U}} C$.

PROPOSITION 2.1.5. *If $A : \mathcal{U}$, the j_A-judgement and the j_A-computation rule imply the following* `IdtoEqv`*-judgement and* `idtoEqv`*-computation rule:*

$$\mathtt{IdtoEqv} : \prod_{X:\mathcal{U}} \prod_{p:X=_{\mathcal{U}}A} X \simeq_{\mathcal{U}} A$$
$$\mathtt{IdtoEqv}(A, \mathtt{refl}_A) \equiv (\mathrm{id}_A, e_A).$$

PROOF. If $C(X, p) \equiv X \simeq_{\mathcal{U}} A$, then $C(A, \mathtt{refl}_A) \equiv A \simeq_{\mathcal{U}} A$. □

The univalence axiom asserts that the function $\mathtt{IdtoEqv}(X) : X =_{\mathcal{U}} A \to X \simeq_{\mathcal{U}} A$ is an equivalence with quasi-inverse the function $\mathtt{ua}(X) : X \simeq_{\mathcal{U}} A \to X =_{\mathcal{U}} A$, hence

$$(X \simeq_{\mathcal{U}} A) \simeq_{\mathcal{U}} (X =_{\mathcal{U}} A).$$

Voevodsky's Axiom of Univalence (`UA`): There are the following `ua`-judgement and the right and left `ua`-computation rules, respectively,

$$\mathtt{ua} : \prod_{X:\mathcal{U}} \prod_{e:X\simeq_{\mathcal{U}}A} X =_{\mathcal{U}} A$$

$$\mathtt{ua}(X, \mathtt{IdtoEqv}(X, p)) = p, \quad p : X =_{\mathcal{U}} A,$$

$$[\mathtt{IdtoEqv}(X, \mathtt{ua}(e))]^*(x) = e^*(x), \quad x : X.$$

Because of the IdtoEqv-computation rule we get the following special case of the right ua-computation rule:

$$\mathtt{ua}(A, (\mathrm{id}_A, e_A)) = \mathtt{refl}_A.$$

In the HoTT-book one finds the simpler writing

$$\mathtt{ua}(\mathrm{id}_A) = \mathtt{refl}_A,$$

$$\mathtt{IdtoEqv}(\mathtt{ua}(f), x) = f(x),$$

where the equivalence e is "identified" with $f \equiv e^*$. With this simplification in writing one shows that

$$\mathtt{ua}(g \circ f) = \mathtt{ua}(f) * \mathtt{ua}(g),$$

$$\mathtt{ua}(f)^{-1} = \mathtt{ua}(f)^{-1}.$$

Note that in the right ua-computation rule, due to the definition of equivalence for $\mathtt{IdtoEqv}(X)$, there is propositional equality, while in the corresponding IdtoEqv-computation rule, as in *all* computation rules of the judgements considered so far, there is judgemental equality involved. As it is mentioned in the Notes of Chapter 2 of the HoTT-book,

> It is also sometimes inconvenient that the theorems of §§2.6-2.13 are only propositional equalities ..., since then we must explicitly mention whenever we pass back and forth across them. One direction of current research in homotopy type theory is to describe a type system in which these rules are *judgemental* equalities ... .

Note also that in the model of cubical type theory, where the ua-judgement is provable, its corresponding computation rule implies

$$\mathtt{ua}(A, (\mathrm{id}_A, e_A)) \equiv \mathtt{refl}_A.$$

The same phenomenon occurs in the theory of higher inductive types, as this is presented in the HoTT-book. The computation rules of the judgements of the elimination axioms involve propositional equality, and again in the model of cubical type theory the provable HITs satisfy the computation rules with judgemental equality.

2.2. A Yoneda-version of the axiom of univalence

Voevodsky's formulation of the univalence axiom, "$\mathtt{IdtoEqv}(X)$ is an equivalence", is at first sight different from what we are used to in Martin-Löf type theory. Of course, the type-theoretic Yoneda lemma, which implies the J-judgement, uses a given pair of quasi-inverses in its formulation too. Maybe, the most striking element of Voevodsky's version is that the $\mathtt{ua}$-computation rulee involve propositional equality rather than judgemental equality.

Here we present a Yoneda-version of the axiom of univalence, according to which the computation rule of the corresponding univalence judgement involves only judgemental equality. Moreover, this approach to univalence reveals its "proximity" to the J-judgement, therefore looks "closer" to Martin-Löf's intentional type theory, although the inspiration of this formulation comes from category theory.

The universe $\mathcal{U}$ can be seen as a category with $\mathrm{Hom}(A, B) \equiv A \simeq_{\mathcal{U}} B$, which is locally small, as $(A \simeq_{\mathcal{U}} B) : \mathcal{U}$, and as in the case of the Yoneda lemma for the type-category A, $\mathcal{U}$ can be seen as a category identical to its opposite. A contravariant functor from $\mathcal{U}$ to $\mathcal{U}$ is again a type family $P : \mathcal{U} \to \mathcal{U}$. Next we define the functor $\mathcal{E}$ from $\mathcal{U}$ to $\mathcal{U}^{\mathcal{U}}$.

Proposition 2.2.1. *Let $e : A \simeq_{\mathcal{U}} B$, $X : \mathcal{U}$, and $\mathcal{E} : \mathcal{U} \to (\mathcal{U} \to \mathcal{U})$ be defined by $A \mapsto \mathcal{E}_A$ and $e \mapsto \mathcal{E}(e)$, where $\mathcal{E}_A : \mathcal{U} \to \mathcal{U}$ and*

$$\mathcal{E}_A(X) \equiv X \simeq_{\mathcal{U}} A,$$

$$\mathcal{E}(e) : \mathrm{Hom}(\mathcal{E}_A, \mathcal{E}_B) \equiv \prod_{X:\mathcal{U}} \prod_{e':X\simeq_{\mathcal{U}} A} X \simeq_{\mathcal{U}} B$$

$$\mathcal{E}(e) \equiv \lambda(X : \mathcal{U}, e' : X \simeq_{\mathcal{U}} A).e \circ e'.$$

Then for every $\epsilon : B \simeq_{\mathcal{U}} C$, we have that

$$\mathcal{E}(\epsilon \circ e) \equiv \mathcal{E}(\epsilon) \circ \mathcal{E}(e),$$

$$\mathcal{E}(\mathrm{id}_A) \equiv 1_{\mathcal{E}_A}.$$

Proof. □

Yoneda-version of the univalence axiom ($\mathtt{Y\text{-}UA}$): Let $P : \mathcal{U} \to \mathcal{U}$ and $A : \mathcal{U}$. There is a pair of quasi-inverses

$$(j, i) : \mathrm{Hom}(\mathcal{E}_A, P) \simeq P(A)$$

i.e., there are the following i-judgement and j-judgement:

$$i : P(A) \to \prod_{X:\mathcal{U}} \prod_{e:X\simeq_{\mathcal{U}}A} P(X)$$

$$j : \left(\prod_{X:\mathcal{U}} \prod_{e:X\simeq_{\mathcal{U}}A} P(X) \right) \to P(A)$$

with the following i-computation rule and j-computation rule:

$$i(u)(A, (\mathrm{id}_A, e_A)) \equiv u, \quad u : P(A),$$

$$j(\sigma) \equiv \sigma(A, (\mathrm{id}_A, e_A)), \quad \sigma : \mathrm{Hom}(\mathcal{E}_A, P).$$

PROPOSITION 2.2.2. *The i-judgement of* `Y-UA` *implies the* `ua`*-judgement i.e., there is*

$$\mathtt{ua}' : \prod_{X:\mathcal{U}} \prod_{e:X\simeq_{\mathcal{U}}A} X =_{\mathcal{U}} A,$$

and moreover

$$\mathtt{ua}'(A, (\mathrm{id}_A, e_A)) \equiv \mathtt{refl}_A.$$

PROOF. Let $P : \mathcal{U} \to \mathcal{U}$ defined by $P(X) \equiv X =_{\mathcal{U}} A$. Since

$$i : A =_{\mathcal{U}} A \to \prod_{X:\mathcal{U}} \prod_{e:X\simeq_{\mathcal{U}}A} X =_{\mathcal{U}} A,$$

we define

$$\mathtt{ua}' \equiv \lambda(X : \mathcal{U}, e : X \simeq_{\mathcal{U}} A).i(\mathtt{refl}_A)(X, e),$$

hence

$$\mathtt{ua}'(A, (\mathrm{id}_A, e_A)) \equiv i(\mathtt{refl}_A)(A, (\mathrm{id}_A, e_A)) \equiv \mathtt{refl}_A.$$

□

PROPOSITION 2.2.3. *If $X : \mathcal{U}$ and $p : X =_{\mathcal{U}} A$, then*

$$\mathtt{ua}'(X, \mathtt{IdtoEqv}(X, p)) = p.$$

PROOF. Define $C(X, p) \equiv \mathtt{ua}'(X, \mathtt{IdtoEqv}(X, p)) = p$. Since

$$\begin{aligned} C(A, \mathtt{refl}_A) &\equiv \mathtt{ua}'(A, \mathtt{IdtoEqv}(A, \mathtt{refl}_A)) = \mathtt{refl}_A \\ &\equiv \mathtt{ua}'(A, (\mathrm{id}_A, e_A)) = \mathtt{refl}_A \\ &\equiv \mathtt{refl}_A = \mathtt{refl}_A, \end{aligned}$$

we use the j_A-judgement. □

PROPOSITION 2.2.4. *The* `ua`*-judgement implies the* i*-judgement of* `Y-UA` *i.e., there is*

$$i' : P(A) \to \prod_{X:\mathcal{U}} \prod_{e:X\simeq_{\mathcal{U}} A} P(X),$$

and moreover

$$i'(u)(A, (\mathrm{id}_A, e_A)) = u, \quad u : P(A).$$

PROOF. Let $u : P(A)$. Since $\mathtt{ua}(X, e) : X =_{\mathcal{U}} A$, we get $\mathtt{ua}(X, e)^{-1} : A =_{\mathcal{U}} X$, and consequently we have that

$$\left[\mathtt{ua}(X, e)^{-1}\right]^P_* : P(A) \to P(X).$$

We define

$$i'(u) \equiv \lambda(X : \mathcal{U}, e : X \simeq_{\mathcal{U}} A).\left[\mathtt{ua}(X, e)^{-1}\right]^P_*(u).$$

Thus,

$$\begin{aligned} i'(u)(A, (\mathrm{id}_A, e_A)) &\equiv \left[\mathtt{ua}(A, (\mathrm{id}_A, e_A))^{-1}\right]^P_*(u) \\ &= \left(\mathtt{refl}_A^{-1}\right)^P_*(u) \\ &\equiv \left(\mathtt{refl}_A\right)^P_*(u) \\ &\equiv \mathrm{id}_{P(A)}(u) \\ &\equiv u. \end{aligned}$$

□

Note that the non-trivial judgement in Voevodsky's axiom of univalence is the `ua`-judgement, while in the Yoneda-version of univalence is the i-judgement. The j-judgement of `Y-UA`, a s a judgement, is immediate, and the `IdtoEqv`-judgement follows from the J-judgement of the equality type.

2.3. A strong Yoneda-version of the axiom of univalence

We add a computation rule to our first Yoneda-version of the axiom of univalence in order to unify the J-judgement and the axiom of univalence. Our aim is to get from the strong Yoneda version of the axiom of univalence the J-judgement that corresponds to it (this is the J_e-judgement in Definition 2.3.5) equipped with a computation rule that involves judgemental equality, and not propositional, as is the case in Voevodsky's form of the axiom.

First we define an obvious generalization of the notion of homotopy.

DEFINITION 2.3.1. Let $A, B : \mathcal{U}$ and $Q : A \to B \to \mathcal{U}$ a type family over A and B (or a relation on A, B). If

$$F, G : \prod_{x:A} \prod_{y:B} Q(x, y),$$

we say that F, G are homotopic, $F \approx B$, if there is

$$H : F \approx B \equiv \prod_{x:A} \prod_{y:B} F(x, y) =_{Q(x,y)} G(x, y).$$

PROPOSITION 2.3.2. *Let $A : \mathcal{U}$ and $P : \mathcal{U} \to \mathcal{U}$. If we fix some*

$$\sigma : \mathrm{Hom}(\mathcal{E}_A, P) \equiv \prod_{X:\mathcal{U}} \prod_{f:X\simeq A} P(X),$$

there is a judgement

$$\mathtt{Happly}_{\mathcal{E},\sigma} : \prod_{\tau\in\mathrm{Hom}(\mathcal{E}_A,P)} \prod_{p:\tau=\sigma} \tau \approx \sigma \equiv$$

$$\equiv \prod_{\tau\in\mathrm{Hom}(\mathcal{E}_A,P)} \prod_{p:\tau=\sigma} \left(\prod_{X:\mathcal{U}} \prod_{f:X\simeq A} \tau(X, f) =_{P(X)} \sigma(X, f) \right)$$

such that

$$\mathtt{Happly}_{\mathcal{E},\sigma}(\sigma, \mathtt{refl}_\sigma) \equiv \lambda(X : \mathcal{U}, f : X \simeq A).\mathtt{refl}_{\sigma(X,f)},$$

PROOF. If $C(\tau, p) \equiv \tau \approx \sigma$, then $C(\sigma, \mathtt{refl}_\sigma) \equiv \sigma \approx \sigma$ and $\lambda(X : \mathcal{U}, f : X \simeq A).\mathtt{refl}_{\sigma(X,f)} : C(\sigma, \mathtt{refl}_\sigma)$. What we want follows immediately by the based J-judgement and computation rule. □

Next we equip the Yoneda-version of UA with an explicit description of the homotopies $i \circ j \sim \mathrm{id}_{\mathrm{Hom}(\mathcal{E}_A,P)}$ and $j \circ i \sim \mathrm{id}_{P(A)}$.

Strong Yoneda-version of the univalence axiom (`sY-UA`): Let $A : \mathcal{U}$ and $P : \mathcal{U} \to \mathcal{U}$. There is a pair of quasi-inverses

$$(j, i) : \mathrm{Hom}(\mathcal{E}_A, P) \simeq P(A)$$

in the following explicit way: there are the i-judgement and j-judgement

$$i : P(A) \to \prod_{X:\mathcal{U}} \prod_{f:X\simeq A} P(X),$$

$$j : \left(\prod_{X:\mathcal{U}} \prod_{f:X\simeq A} P(X) \right) \to P(A),$$

equipped with the following i-computation rule and j-computation rule:

$$i(u)(A, \mathrm{id}_A) \equiv u, \quad u : P(A),$$
$$j(\sigma) \equiv \sigma(A, \mathrm{id}_A), \quad \sigma : \mathrm{Hom}(\mathcal{E}_A, P).$$

Moreover, there are the G-judgement and H-judgement:

$$G : i \circ j \sim \mathrm{id}_{\mathrm{Hom}(\mathcal{E}_A,P)} \equiv \prod_{\sigma\in\mathrm{Hom}(\mathcal{E}_A,P)} i(j(\sigma)) = \sigma,$$

$$H : j \circ i \sim \mathrm{id}_{P(A)} \equiv \prod_{u:P(A)} j(i(u)) = u,$$

equipped with the following G-computation rule and H-computation rule:

$$\mathtt{Happly}_{\mathcal{E},\sigma}\big(i(j(\sigma)), G(\sigma)\big)(A, \mathrm{id}_A) \equiv \mathtt{refl}_{\sigma(A,\mathrm{id}_A)}, \quad \sigma : \mathrm{Hom}(\mathcal{E}_A, P),$$
$$H(u) \equiv \mathtt{refl}_u, \quad u : P(A).$$

The last two computation rules, which make the difference between the two Yoneda-versions of UA, are justified as follows:
Since

$$G(\sigma) : i(j(\sigma)) = \sigma,$$

we have that

$$\mathtt{Happly}_{\mathcal{E},\sigma}\big(i(j(\sigma)), G(\sigma)\big) : \prod_{X:\mathcal{U}} \prod_{f:X\simeq A} i(j(\sigma))(X, f) =_{P(X)} \sigma(X, f),$$

$$\mathtt{Happly}_{\mathcal{E},\sigma}\big(i(j(\sigma)), G(\sigma)\big)(A, \mathrm{id}_A) : i(j(\sigma))(A, \mathrm{id}_A) =_{P(A)} \sigma(A, \mathrm{id}_A).$$

By the j, i-computation rules we have that

$$i(j(\sigma))(A, \mathrm{id}_A) \equiv i(\sigma(A, \mathrm{id}_A))(A, \mathrm{id}_A) \equiv \sigma(A, \mathrm{id}_A),$$

therefore

$$\mathtt{Happly}_{\mathcal{E},\sigma}\big(i(j(\sigma)), G(\sigma)\big)(A, \mathrm{id}_A) : \sigma(A, \mathrm{id}_A) =_{P(A)} \sigma(A, \mathrm{id}_A).$$

Similarly, if $u : P(A)$,

$$H(u) : j(i(u)) = u,$$

and since

$$j(i(u)) \equiv i(u)(A, \mathrm{id}_A) \equiv u,$$

we get

$$H(u) : u =_{P(A)} u.$$

It is natural to demand, as is the case in all axioms of this kind, like the J-axiom, that the terms associated to these computation rules are the most expected ones. Note though, that the H-computation rule is not significant, while the G-computation rule makes the whole difference between the two formulations of the Yoneda-version of univalence. Next follows the (strong) analogue to Lemma 1.6.7.

LEMMA 2.3.3. *If $B : \mathcal{U}$, the strong Yoneda-judgements and the corresponding computation rules imply the following judgement and computation rules:*

$$(j_B, i_B) : \left(\prod_{X:\mathcal{U}} \prod_{f:X\simeq A} B\right) \simeq B$$

$$i_B : B \to \prod_{X:\mathcal{U}} \prod_{f:X\simeq A} B,$$

$$j_B : \left(\prod_{X:\mathcal{U}} \prod_{f:X\simeq A} B\right) \to B,$$

$$i_B(b)(A, \mathrm{id}_A) \equiv b, \quad b : B,$$

$$j_B(\sigma) \equiv \sigma(A, \mathrm{id}_A), \quad \sigma : \prod_{X:\mathcal{U}} \prod_{f:X\simeq A} B,$$

$$G_B : \prod_{\sigma\in \mathrm{Hom}(\mathcal{E}_A,B)} i_B(j_B(\sigma)) = \sigma,$$

$$H_B : \prod_{b:B} j(i(b)) = b,$$

$$\mathtt{Happly}_{\mathcal{E},\sigma}\Big(i_B(j_B(\sigma)), G_B(\sigma)\Big)(A, \mathrm{id}_A) \equiv \mathtt{refl}_{\sigma(A,\mathrm{id}_A)}, \quad \sigma : \mathrm{Hom}(\mathcal{E}_A, B),$$

$$H_B(b) \equiv \mathtt{refl}_b, \quad b : B.$$

Moreover, if $b : B$, $X : \mathcal{U}$ and $f : X \simeq A$, then, if

$$\big[\sigma_b \equiv \lambda(X : \mathcal{U}, f : X \simeq A).b\big] : \prod_{X:\mathcal{U}} \prod_{f:X\simeq A} B,$$

we have that

$$\mathtt{Happly}_{\mathcal{E},\sigma_b}\Big(i_B(j_B(\sigma_b)), G_B(\sigma_b)\Big)(X,f) : \big[i_B(b)(X,f) =_B b\big],$$

such that

$$\mathtt{Happly}_{\mathcal{E},\sigma_b}\Big(i_B(j_B(\sigma_b)), G_B(\sigma_b)\Big)(A,\mathrm{id}_A) \equiv \mathtt{refl}_b.$$

Proof. We use the strong Yoneda-judgement and computation rules for the constant type family $P : \mathcal{U} \to \mathcal{U}$, defined by $P(X) \equiv B$, for every $X : \mathcal{U}$. For the propositional equality

$$i_B(b)(X,f) =_B b$$

we work as follows. Since

$$i_B(j_B(\sigma_b)) \equiv i_B(\sigma_b(A,\mathrm{id}_A)) \equiv i_B(b),$$

we get

$$G_B(\sigma_b) : i_B(b) = \sigma_b.$$

Moreover,

$$\mathtt{Happly}_{\mathcal{E},\sigma_b}\Big(i_B(j_B(\sigma_b)), G_B(\sigma_b)\Big) : \prod_{X:\mathcal{U}} \prod_{f:X\simeq A} i_B(j_B(\sigma_b))(X,f) =_B \sigma_b(X,f),$$

hence

$$\begin{aligned}\mathtt{Happly}_{\mathcal{E},\sigma_b}\Big(i_B(j_B(\sigma_b)), G_B(\sigma_b)\Big)(X,f) : i_B(b)(X,f) &=_B \sigma_b(X,f)\\ &\equiv i_B(b)(X,f) =_B b.\end{aligned}$$

Consequently,

$$\begin{aligned}\mathtt{Happly}_{\mathcal{E},\sigma_b}\Big(i_B(j_B(\sigma_b)), G_B(\sigma_b)\Big)(A,\mathrm{id}_A) : i_B(b)(A,\mathrm{id}_A) &=_B \sigma_b(A,\mathrm{id}_A)\\ &\equiv b =_B b,\end{aligned}$$

and by the G-computation rule of the strong Yoneda-version of UA we get

$$\mathtt{Happly}_{\mathcal{E},\sigma_b}\Big(i_B(j_B(\sigma_b)), G_B(\sigma_b)\Big)(A,\mathrm{id}_A) \equiv \mathtt{refl}_{\sigma_b(A,\mathrm{id}_A)} \equiv \mathtt{refl}_b.$$

□

The next corollary is the (strong) analogue to Corollary 1.6.8, where the corresponding M-computation is also proved.

COROLLARY 2.3.4. *If*

$$E_A \equiv \sum_{X:\mathcal{U}} X \simeq A,$$

the judgements and computational rules of the strong Yoneda-version of UA imply the following M_e-judgement and M_e-computation rule:

$$M_e : \prod_{X:\mathcal{U}} \prod_{f:X\simeq A} (X, f) =_{E_A} (A, \mathrm{id}_A),$$

$$M_e(A, \mathrm{id}_A) \equiv \mathtt{refl}_{(A,\mathrm{id}_A)}.$$

PROOF. By Lemma 2.3.3 we have that there is a pair of quasi-inverses

$$(j_{E_A}, i_{E_A}) : \left(\prod_{X:\mathcal{U}} \prod_{f:X\simeq A} E_A\right) \simeq E_A$$

with the associated judgements and computation rules determined in the formulation of Lemma 2.3.3. Let $\tau : \prod_{X:\mathcal{U}} \prod_{f:X\simeq A} E_A$, where

$$\tau \equiv \lambda(X : \mathcal{U}, f : X \simeq A).(X, f).$$

We have that

$$\begin{aligned}(X, f) &\equiv \tau(X, f)\\ &=_{E_A} \left[i_{E_A}(j_{E_A}(\tau))\right](X, f)\\ &\equiv \left[i_{E_A}(\tau(A, \mathrm{id}_A))\right](X, f)\\ &\equiv \left[i_{E_A}((A, \mathrm{id}_A))\right](X, f)\\ &=_{E_A} (A, \mathrm{id}_A).\end{aligned}$$

Since

$$\mathtt{Happly}_{\mathcal{E},\tau}\left(i_{E_A}(j_{E_A}(\tau)), G_{E_A}(\tau)\right)(X, f) : \left[i_{E_A}(j_{E_A}(\tau))\right](X, f) = \tau(X, f),$$

we get that the term

$$\left[\mathtt{Happly}_{\mathcal{E},\tau}\left(i_{E_A}(j_{E_A}(\tau)), G_{E_A}(\tau)\right)(X, f)\right]^{-1}$$

is of type

$$\tau(X, f) = \left[i_{E_A}(j_{E_A}(\tau))\right](X, f).$$

Since

$$\mathtt{Happly}_{\mathcal{E},\sigma_{(A,\mathrm{id}_A)}}\left(i_{E_A}(j_{E_A}(\sigma_{(A,\mathrm{id}_A)})), G_{E_A}(\sigma_{(A,\mathrm{id}_A)})\right)(X, f)$$

is a term of type

$$\big[i_{E_A}((A,\mathrm{id}_A))\big](X,f) =_{E_A} (A,\mathrm{id}_A),$$

hence we have determined the terms witnessing the two equalities in between the equality $(X,f) =_{E_A} (A,\mathrm{id}_A)$, and we define

$$\begin{aligned} M_e(X,f) \equiv &\left[\mathtt{Happly}_{\mathcal{E},\tau}\left(i_{E_A}(j_{E_A}(\tau)), G_{E_A}(\tau)\right)(X,f)\right]^{-1} * \\ &\mathtt{Happly}_{\mathcal{E},\sigma_{(A,\mathrm{id}_A)}}\left(i_{E_A}(j_{E_A}(\sigma_{(A,\mathrm{id}_A)})), G_{E_A}(\sigma_{(A,\mathrm{id}_A)})\right)(X,f). \end{aligned}$$

Consequently,

$$\begin{aligned} M_e(A,\mathrm{id}_A) \equiv &\left[\mathtt{Happly}_{\mathcal{E},\tau}\left(i_{E_A}(j_{E_A}(\tau)), G_{E_A}(\tau)\right)(A,\mathrm{id}_A)\right]^{-1} * \\ &\mathtt{Happly}_{\mathcal{E},\sigma_{(A,\mathrm{id}_A)}}\left(i_{E_A}(j_{E_A}(\sigma_{(A,\mathrm{id}_A)})), G_{E_A}(\sigma_{(A,\mathrm{id}_A)})\right)(A,\mathrm{id}_A) \\ &\equiv \left[\mathtt{refl}_{\tau(A,\mathrm{id}_A)}\right]^{-1} * \mathtt{refl}_{(A,\mathrm{id}_A)} \\ &\equiv \left[\mathtt{refl}_{(A,\mathrm{id}_A)}\right]^{-1} * \mathtt{refl}_{(A,\mathrm{id}_A)} \\ &\equiv \mathtt{refl}_{(A,\mathrm{id}_A)} * \mathtt{refl}_{(A,\mathrm{id}_A)} \\ &\equiv \mathtt{refl}_{(A,\mathrm{id}_A)}. \end{aligned}$$

□

Next we describe the based J-rule that corresponds to the strong Yoneda-version of UA. Note that the Yoneda-version of UA implies the corresponding judgement, but we need the strong version to get its computation rule.

Definition 2.3.5. We call the following judgment and computation rule

$$J_e : \prod_{C:\prod_{X:\mathcal{U}}\prod_{f:X\simeq A}\mathcal{U}} \prod_{c:C(A,\mathrm{id}_A)} \left(\prod_{X:\mathcal{U}}\prod_{f:X\simeq A} C(X,f)\right)$$

$$J_e(C,c,A,\mathrm{id}_A) \equiv c$$

the `Eq-J`-judgement and the `Eq-J`-computation rule, respectively.

The next theorem is the (strong) analogue to Theorem 1.6.9.

Theorem 2.3.6. *The judgements and computational rules of the strong Yoneda-version of UA imply the* `Eq-J`*-judgement and the* `Eq-J`*-computation rule.*

PROOF. We fix $C : \prod_{X:\mathcal{U}} \prod_{f:X\simeq A} \mathcal{U}$ and $c \in C(A, \mathrm{id}_A)$. Let $E_A \equiv \sum_{X:\mathcal{U}} X \simeq A$, and $P : E_A \to \mathcal{U}$, defined by

$$P((X, f)) \equiv C(X, f),$$

for every $X : \mathcal{U}$ and $f : X \simeq A$. By Corollary 2.3.4

$$M_e(X, f) : (X, f) =_{E_A} (A, \mathrm{id}_A),$$

hence

$$M_e(X, f)^{-1} : (A, \mathrm{id}_A) =_{E_A} (X, f).$$

Consequently

$$\left[M_e(X, f)^{-1}\right]^P_* : P((A, \mathrm{id}_A)) \to P((X, f)) \equiv C(A, \mathrm{id}_A) \to C(X, f).$$

We define

$$J_e(C, c, X, f) \equiv \left[M_e(X, f)^{-1}\right]^P_*(c).$$

By Corollary 2.3.4 we get

$$\begin{aligned} J_e(C, c, A, \mathrm{id}_A) &\equiv \left[M_e(A, \mathrm{id}_A)^{-1}\right]^P_*(c) \\ &\equiv \left[(\mathtt{refl}_{(A,\mathrm{id}_A)})^{-1}\right]^P_*(c) \\ &\equiv \left[\mathtt{refl}_{(A,\mathrm{id}_A)}\right]^P_*(c) \\ &\equiv \mathrm{id}_{P((A,\mathrm{id}_A))}(c) \\ &\equiv \mathrm{id}_{C(A,\mathrm{id}_A)}(c) \\ &\equiv c. \end{aligned}$$

□

One can show that the J_e-judgement implies the judgments of the strong Yoneda version of univalence. We need to check if the same holds for the corresponding computation rules. The `Eq-J`-judgement and `Eq-J`-rule form the inductive, or "type-theoretic" version of univalence, while the strong Yoneda version can be seen as the "categorical" version of univalence axiom. In the HoTT-book (Corollary 5.8.5) the same judgement follows from univalence but with the computation rule involving propositional equality.

Next we show that the univalence function

$$\mathtt{ua}' \equiv \lambda(X : \mathcal{U}, f : X \simeq A).i(\mathtt{refl}_A)(X, f)$$

defined in the proof of Proposition 2.2.2, and for which we know that

$$\mathtt{ua}'(A, \mathrm{id}_A) \equiv \mathtt{refl}_A,$$

satisfies in the context of the strong Yoneda-version of univalence also the second computation rule of Voevodsky's univalence axiom.

COROLLARY 2.3.7. *If* $f : X \simeq A$, *then*

$$\mathtt{IdtoEqv}(X, \mathtt{ua}'(X, f)) = f.$$

PROOF. We define $C(X, f) \equiv \mathtt{IdtoEqv}(X, \mathtt{ua}'(X, f)) = f$. Since

$$\begin{aligned} C(A, \mathrm{id}_A) &\equiv \mathtt{IdtoEqv}(A, \mathtt{ua}'(A, \mathrm{id}_A)) = \mathrm{id}_A \\ &\equiv \mathtt{IdtoEqv}(A, \mathtt{refl}_A) = \mathrm{id}_A \\ &\equiv \mathrm{id}_A = \mathrm{id}_A, \end{aligned}$$

we have that $\mathtt{refl}_{\mathrm{id}_A} : C(A, \mathrm{id}_A)$, and we use Theorem 2.3.6. □

CHAPTER 3

Equivalence relations on types

3.1. Setoids

Bishop's notion of set, introduced in [**5**], is interpreted in Martin-Löf's type theory (see [**10**] and [**11**]) through the notion of setoid (see e.g., [**12**] and [**4**]). Here, following the HoTT-book, we define setoids through the notion of a mere proposition and not, as is standard in the older literature of type theory, through a universe $\mathtt{U}_0$ of propositions (see e.g., [**7**]).

Definition 3.1.1. A type A is a *mere proposition*, if the following type is inhabited

$$\mathtt{isProp}(A) \equiv \prod_{x,y:B} (x =_A y).$$

Definition 3.1.2. A type B is a *proposition*, if the following type is inhabited

$$\mathtt{isProp}(B) \equiv \prod_{x,y:B} (x =_B y).$$

$$\sim_A : A \to A \to \mathcal{U}$$
$$\mathtt{isProp}(x \sim_A y)$$
$$\prod_{x,y:A} \prod_{e:x\sim_A y} f(x) \sim_B f(y).$$
$$(x,y) \simeq_{A\times B} (x',y') \equiv (x \simeq_A x') \times (y \simeq_B y')$$
$$B^A \equiv \sum_{f:A\to B} \prod_{x,y:A} (x \sim_A y \to f(x) \sim_B f(y))$$
$$(f,u) \sim_{B^A} (g,w) \equiv \prod_{x:A} (f(x) = g(x))$$

Setoids and setoid functions form a cartesian closed category.
We can realize function extensionality in ITT via the setoid B^A.

3.2. Typoids

DEFINITION 3.2.1. A structure $\mathcal{A} \equiv (A, \simeq_{\mathcal{A}}, \mathtt{eqv}_{\mathcal{A}}, *_{\mathcal{A}}, ^{-1_{\mathcal{A}}}, \cong_{\mathcal{A}})$ is called a 2-*typoid*, or simpler here a *typoid*, if $A : \mathcal{U}$ and $\simeq_{\mathcal{A}} : \prod_{x,y:A} \mathcal{U}$ is an equivalence relation on A such that

$$\mathtt{eqv}_{\mathcal{A}} : \prod_{x:A} (x \simeq_{\mathcal{A}} x),$$

$$*_{\mathcal{A}} : \prod_{x,y,z:A} \prod_{e:x\simeq_{\mathcal{A}} y} \prod_{d:y\simeq_{\mathcal{A}} z} x \simeq_{\mathcal{A}} z,$$

$$^{-1_{\mathcal{A}}} : \prod_{x,y:A} \prod_{e:x\simeq_{\mathcal{A}} y} y \simeq_{\mathcal{A}} x$$

and

$$\cong_{\mathcal{A}} : \prod_{x,y:A} \prod_{e,d:x\simeq_{\mathcal{A}} y} \mathcal{U}$$

such that

$$\cong_{\mathcal{A}} (x, y) : \prod_{e,d:x\simeq_{\mathcal{A}} y} \mathcal{U}$$

is an equivalence relation on $x \simeq_{\mathcal{A}} y$, for every $x, y : A$, and the following conditions are satisfied:

(Typ$_1$) $(\mathtt{eqv}_x *_{\mathcal{A}} e) \cong_{\mathcal{A}} e$ and $(e *_{\mathcal{A}} \mathtt{eqv}_y) \cong_{\mathcal{A}} e$.

(Typ$_2$) $(e *_{\mathcal{A}} e^{-1_{\mathcal{A}}}) \cong_{\mathcal{A}} \mathtt{eqv}_x$ and $(e^{-1_{\mathcal{A}}} *_{\mathcal{A}} e) \cong_{\mathcal{A}} \mathtt{eqv}_y$.

(Typ$_3$) $(e_1 *_{\mathcal{A}} e_2) *_{\mathcal{A}} e_3 \cong_{\mathcal{A}} e_1 *_{\mathcal{A}} (e_2 *_{\mathcal{A}} e_3)$.

(Typ$_4$) $e_1 \cong_{\mathcal{A}} d_1 \to e_2 \cong_{\mathcal{A}} d_2 \to (e_1 *_{\mathcal{A}} e_2) \cong_{\mathcal{A}} (d_1 *_{\mathcal{A}} d_2)$.

A *pretypoid* is a pair $(A, \simeq_{\mathcal{A}})$, where $\simeq_{\mathcal{A}}$ is an equivalence relation on A.

One could write the notion of typoid as the following type:

$$\begin{aligned}\mathtt{Typoid}(\mathcal{A}) \equiv \sum_{A:\mathcal{U}} \sum_{\simeq_{\mathcal{A}}:\prod_{x,y:A}\mathcal{U}} \sum_{\mathtt{eqv}_{\mathcal{A}}:\prod_{x:A}(x\simeq_{\mathcal{A}}x)} \\ \sum_{*_{\mathcal{A}}:\prod_{x,y,z:A}\prod_{e:x\simeq_{\mathcal{A}}y}\prod_{d:y\simeq_{\mathcal{A}}z} x\simeq_{\mathcal{A}}z} \sum_{^{-1_{\mathcal{A}}}:\prod_{x,y:A}\prod_{e:x\simeq_{\mathcal{A}}y} y\simeq_{\mathcal{A}}x} \\ \sum_{\cong_{\mathcal{A}}:\prod_{x,y:A}\prod_{e,d:x\simeq_{\mathcal{A}}y}} \Big(\mathrm{Typ}_1{}' \times \mathrm{Typ}_2{}' \times \mathrm{Typ}_3{}' \times \mathrm{Typ}_4{}'\Big).\end{aligned}$$

where e.g., $\mathrm{Typ}_1{}'$ is

$$\prod_{x,y:A} \prod_{e:x\simeq_{\mathcal{A}} y} (\mathtt{eqv}_x *_{\mathcal{A}} e) \cong_{\mathcal{A}} e.$$

When it is clear from the context we may omit the subscripts.

PROPOSITION 3.2.2. *Let $\mathcal{A}$ be a typoid, $x, y : A$, and $e, d : x \simeq y$.*
(i) $(\mathtt{eqv}_x)^{-1} \cong \mathtt{eqv}_x$.
(iii) $(e^{-1})^{-1} \cong e$.
(iii) $e \cong d \to e^{-1} \cong d^{-1}$.

PROOF. (i) By Typ_2 we have that $\mathtt{eqv}_x * \mathtt{eqv}_x{}^{-1} \cong \mathtt{eqv}_x$ and by Typ_1 we also have $\mathtt{eqv}_x * \mathtt{eqv}_x{}^{-1} \cong \mathtt{eqv}_x{}^{-1}$, and eventually $(\mathtt{eqv}_x)^{-1} \cong \mathtt{eqv}_x$.
(ii) Since $(e^{-1})^{-1} * e^{-1} \cong \mathtt{eqv}_x$, by Typ_4 we get $((e^{-1})^{-1} * e^{-1}) * e \cong \mathtt{eqv}_x * e$, and consequently $(e^{-1})^{-1} * \mathtt{eqv}_y \cong e$, hence $(e^{-1})^{-1} \cong e$.
(iii) By condition (iv) we get $e^{-1} * e \cong e^{-1} * d$, hence $e^{-1} * d \cong \mathtt{eqv}_y$, therefore $(e^{-1} * d) * d^{-1} \cong \mathtt{eqv}_y * d^{-1}$ i.e., $e^{-1} * \mathtt{eqv}_x \cong d^{-1}$, hence $e^{-1} \cong d^{-1}$. □

EXAMPLE 3.2.3. Using basic properties of equality $p =_{x=_A y} q$, of concatenation $p * q$ and inversion p^{-1} of paths it is easy to see that the structure

$$\mathcal{A}_0 \equiv (A, =_A, \mathtt{refl}_A, *, ^{-1}, \cong_{\mathcal{A}_0})$$

is a typoid, where $\cong_{\mathcal{A}_0} : \prod_{x,y:A} \prod_{e,e':x=_A y} \mathcal{U}$ is defined by

$$\cong_{\mathcal{A}_0} (x, y, e, e') \equiv (e =_{x=_A y} e'),$$

for every $x, y : A$ and $e, e' : x =_A y$. We call $\mathcal{A}_0$ the *equality* typoid, and its typoid structure the *equality* typoid structure on A.

EXAMPLE 3.2.4. If $A, B : \mathcal{U}$, it is easy to see that the structure

$$\mathrm{Fun}(A, B) \equiv (A \to B, \simeq_{A\to B}, \mathtt{eqv}_{A\to B}, *_{A\to B}, ^{-1_{A\to B}}, \cong_{A\to B})$$

is a typoid, where

$$f \simeq_{A\to B} g \equiv \prod_{x:A} f(x) =_B g(x),$$

while if $H, H' : f \simeq_{A\to B} g$ and $G : g \simeq_{A\to B} h$, we define

$$H *_{A\to B} G \equiv \lambda(x : A).(H(x) * G(x)),$$
$$H^{-1_{A\to B}} \equiv \lambda(x : A).(H(x))^{-1},$$
$$\mathtt{eqv}_f \equiv \lambda(x : A).\mathtt{refl}_{f(x)},$$
$$H \cong_{A\to B} H' \equiv \prod_{x:A} H(x) =_{(f(x)=_B g(x))} H'(x).$$

We call is the *typoid of functions* Similarly, we define a typoid structure on the type of dependent functions $\prod_{x:A} P(x)$, where $P : A \to \mathcal{U}$ is a type family over A.

Example 3.2.5. Using Voevodsky's definition $\mathtt{isequiv}(f)$, it is easy to show that

$$\mathrm{Uni} \equiv (\mathcal{U}, \simeq_{\mathcal{U}}, \mathtt{eqv}_{\mathcal{U}}, *_{\mathcal{U}}, ^{-1_{\mathcal{U}}}, \cong_{\mathcal{U}})$$

is a typoid, where

$$A \simeq_{\mathrm{Uni}} B \equiv \sum_{f:A\to B} \mathtt{isequiv}(f),$$

while if $(f, u), (f', u') : A \simeq_{\mathrm{Uni}} B$ and $(g, v) : B \simeq_{\mathrm{Uni}} C$, we define

$$(f, u) *_{\mathrm{Uni}} (g, v) \equiv (g \circ f, w),$$
$$(f, u)^{-1_{\mathrm{Uni}}} \equiv (f^{-1}, u^{-1}),$$
$$\mathtt{eqv}_A \equiv (\mathrm{id}_A, i),$$
$$(f, u) \cong_{\mathrm{Uni}} (f', u') \equiv \prod_{x:A} f(x) =_B f'(x),$$

where $w : \mathtt{isequiv}(g \circ f), u^{-1} : \mathtt{isequiv}(f^{-1})$ and $i : \mathtt{isequiv}(\mathrm{id}_A)$. Note that the definition of $(f, u) \cong_{\mathrm{Uni}} (f', u')$ is based on the fact that all terms of type $\mathtt{isequiv}(f)$ are equal. We call Uni the *universal typoid*.

From now on $\mathcal{A}, \mathcal{B}$ denote typoids, i.e., $\mathcal{A} \equiv (A, \simeq_{\mathcal{A}}, \mathtt{eqv}_{\mathcal{A}}, *_{\mathcal{A}}, ^{-1_{\mathcal{A}}}, \cong_A)$ and $\mathcal{B} \equiv (B, \simeq_{\mathcal{B}}, \mathtt{eqv}_{\mathcal{B}}, *_{\mathcal{B}}, ^{-1_{\mathcal{B}}}, \cong_{\mathcal{B}})$.

Definition 3.2.6. If $\mathcal{A}, \mathcal{B}$ are typoids, we call a function $f : A \to B$ a *typoid function*, if there are dependent functions

$$\Phi_f : \prod_{x,y:A} \prod_{e:x\simeq_{\mathcal{A}} y} f(x) \simeq_{\mathcal{B}} f(y),$$

$$\Phi^2_f : \prod_{x,y:A} \prod_{e,d:x\simeq_{\mathcal{A}} y} \prod_{i:e\cong_{\mathcal{A}} d} \Phi_f(x, y, e) \cong_{\mathcal{B}} \Phi_f(x, y, d),$$

which we call an 1-*associate* of f and a 2-*associate* of f with respect to Φ_f, respectively, such that for every $x, y, z : A$ and every $e_1 : x \simeq_A y$, $e_2 : y \simeq_A z$ the following types are inhabited:

(*i*) $\Phi_f(x, x, \mathtt{eqv}_x) \cong_{\mathcal{B}} \mathtt{eqv}_{f(x)}$,

(*ii*) $\Phi_f(x, z, e_1 *_A e_2) \cong_{\mathcal{B}} \Phi_f(x, y, e_1) *_{\mathcal{B}} \Phi_f(y, z, e_2)$.

If $\Phi_f(x, x, \mathtt{eqv}_x) \equiv \mathtt{eqv}_{f(x)}$, for every $x : A$, we call f *strict* with respect to Φ_f. If $\mathcal{A}, \mathcal{B}$ are pretypoids, we call a function $f : A \to B$ a *pretypoid function*, if there is an 1-*associate* of f.

The 1-*associate* Φ_f of f witnesses that f preserves the equivalences between points, as

$$\Phi_f(x,y) : x \simeq_{\mathcal{A}} y \to f(x) \simeq_{\mathcal{B}} f(y),$$

while the 2-*associate* Φ^2_f of f with respect to Φ_f witnesses that Φ_f preserves the equivalences between equivalences, as

$$\Phi^2_f(x,y,e,d) : e \cong_{\mathcal{A}} d \to \Phi_f(x,y,e) \cong_{\mathcal{B}} \Phi_f(x,y,d).$$

PROPOSITION 3.2.7. *If $\mathcal{A},\mathcal{B}$ are typoids and $f : A \to B$ is a typoid function, then, for every $x,y : A$ and $e : x \simeq_{\mathcal{A}} y$, then*

$$\Phi_f(y,x,e^{-1_{\mathcal{A}}}) \cong_{\mathcal{B}} [\Phi_f(x,y,e)]^{-1_{\mathcal{B}}}.$$

PROOF. Since $(e *_{\mathcal{A}} e^{-1_{\mathcal{A}}}) \cong_{\mathcal{A}} \mathtt{eqv}_x$, by the existence of Φ^2_f we get a term of type $\Phi_f(x,x,e *_{\mathcal{A}} e^{-1_{\mathcal{A}}}) \cong_{\mathcal{B}} \Phi_f(x,x,\mathtt{eqv}_x)$, hence a term of type $(\Phi_f(x,y,e) *_{\mathcal{B}} \Phi_f(y,x,e^{-1_{\mathcal{A}}})) \cong_{\mathcal{B}} \mathtt{eqv}_{f(x)}$. By condition Typ_4 we get $[\Phi_f(x,y,e)]^{-1_{\mathcal{B}}} *_{\mathcal{B}} (\Phi_f(x,y,e) *_{\mathcal{B}} \Phi_f(y,x,e^{-1_{\mathcal{A}}})) \cong_{\mathcal{B}} [\Phi_f(x,y,e)]^{-1_{\mathcal{B}}} *_{\mathcal{B}} \mathtt{eqv}_{f(x)}$, therefore the required type is inhabited. □

EXAMPLE 3.2.8. If $\mathcal{A}_0,\mathcal{B}_0$ are equality typoids and $f : A \to B$, then f is a strict typoid function with respect to its 1-associate, the application function $\mathtt{ap}_f$, and with 2-associate with respect to $\mathtt{ap}_f$ the two-dimensional application function $\mathtt{ap}^2_f$ of f, where

$$\mathtt{ap}_f : \prod_{x,y:A} \prod_{p:x=_A y} f(x) =_B f(y),$$

$$\mathtt{ap}^2_f : \prod_{x,y:A} \prod_{p,q:x=_A y} \prod_{r:p=_{(x=_A y)} q} \mathtt{ap}_f(x,y,p) =_{(f(x)=_B f(y))} \mathtt{ap}_f(x,y,q).$$

The properties $\mathtt{ap}_f(x,x,\mathtt{refl}_x) \equiv \mathtt{refl}_{f(x)}$ and $\mathtt{ap}_f(x,z,p{*}q) = \mathtt{ap}_f(x,y,p){*}\mathtt{ap}_f(y,z,q)$ follow immediately from path induction (see section 2.2 of [**16**]).

PROPOSITION 3.2.9. *If $\mathcal{A},\mathcal{B},\mathcal{C}$ are typoids and $f : A \to B, g : B \to C$ are typoid functions with associates Φ_f, Φ^2_f and Φ_g, Φ^2_g, respectively, then $g \circ f : A \to C$ is a typoid function with associates*

$$\Phi_{g\circ f} : \prod_{x,y:A} \prod_{e:x\simeq_{\mathcal{A}} y} g(f(x)) \simeq_C g(f(y)),$$

$$\Phi^2_{g\circ f} : \prod_{x,y:A} \prod_{e,d:x\simeq_{\mathcal{A}} y} \prod_{i:e\cong_{\mathcal{A}} d} \Phi_{g\circ f}(x,y,e) \cong_C \Phi_{g\circ f}(x,y,d),$$

defined for every $x, y : A, e, d : x \simeq_{\mathcal{A}} y, i : e \cong_{\mathcal{A}} d$ *by*

$$\Phi_{g\circ f}(x, y, e) \equiv \Phi_g\Big(f(x), f(y), \Phi_f(x, y, e)\Big),$$

$$\Phi^2_{g\circ f}(x, y, e, d, i) \equiv \Phi^2_g\Big(f(x), f(y), \Phi_f(x, y, e), \Phi_f(x, y, d), \Phi^2_f(x, y, e, d, i)\Big).$$

If f, g *are strict with respect to* Φ_f, Φ_g, $g \circ f$ *is strict with respect to* $\Phi_{g\circ f}$.

Proof. If $j : \Phi_f(x, x, \mathtt{eqv}_x) \cong_{\mathcal{B}} \mathtt{eqv}_{f(x)}$, then the term

$$\Phi^2_g(f(x), f(x), \Phi_f(x, x, \mathtt{eqv}_x), \mathtt{eqv}_{f(x)})$$

is of type $\Phi_g(f(x), f(x), \Phi_f(x, x, \mathtt{eqv}_x)) \cong_C \Phi_g(f(x), f(x), \mathtt{eqv}_{f(x)})$. Since $\Phi_g(f(x), f(x), \mathtt{eqv}_{f(x)}) \cong_C \mathtt{eqv}_{g(f(x))}$, we define

$$\Phi_{g\circ f}(x, x, \mathtt{eqv}_x) \equiv \Phi_g(f(x), f(x), \Phi_f(x, x, \mathtt{eqv}_x)) \cong_C \mathtt{eqv}_{g(f(x))}.$$

Using the existence of Φ^2_g we get the following $\cong_C$-equivalences

$$\begin{aligned}
\Phi_{g\circ f}(x, z, e *_{\mathcal{A}} d) &\equiv \Phi_g(f(x), f(z), \Phi_f(x, z, e *_{\mathcal{A}} d)) \\
&\cong \Phi_g(f(x), f(z), \Phi_f(x, y, e) *_{\mathcal{B}} \Phi_f(y, z, d)) \\
&\cong \Phi_g(f(x), f(y), \Phi_f(x, y, e)) *_C \Phi_g(f(y), f(z), \Phi_f(y, z, d)) \\
&\equiv \Phi_{g\circ f}(x, y, e) *_C \Phi_{g\circ f}(y, z, d).
\end{aligned}$$

Since $\Phi^2_f(x, y, e, d, i) : \Phi(x, y, e) \cong_{\mathcal{B}} \Phi_f(x, y, d)$, the term

$$\Phi^2_g(f(x), f(y), \Phi(x, y, e), \Phi_f(x, y, d), \Phi^2_f(x, y, e, d, i))$$

is of type $\Phi_g(f(x), f(y), \Phi_f(x, y, e)) \cong_C \Phi_g(f(x), f(y), \Phi_f(x, y, d))$ i.e., of type $\Phi_{g\circ f}(x, y, e) \cong_C \Phi_{g\circ f}(x, y, d)$, hence $\Phi^2_{g\circ f}$ is well-defined. For the preservation of strictness we have that

$$\begin{aligned}
\Phi_{g\circ f}(x, x, \mathtt{eqv}_x) &\equiv \Phi_g(f(x), f(x), \Phi_f(x, x, \mathtt{eqv}_x)) \\
&\equiv \Phi_g(f(x), f(x), \mathtt{eqv}_{f(x)}) \\
&\equiv \mathtt{eqv}_{g(f(x))}.
\end{aligned}$$

□

Proposition 3.2.10. *If* $\mathcal{A}$ *is a typoid, the identity* $\mathrm{id}_A : A \to A$ *is a strict typoid function from* $\mathcal{A}_0$ *to* $\mathcal{A}$ *with respect to its* 1*-associate*

$$\mathtt{idtoEqv}_{\mathcal{A}} : \prod_{x,y:A} \prod_{p:x=_A y} x \simeq_{\mathcal{A}} y,$$

$$\mathtt{idtoEqv}_{\mathcal{A}}(x, y, p) \equiv p^{P_x}_*(\mathtt{eqv}_x),$$

where $P_x : A \to \mathcal{U}$ *is defined by* $P_x(z) \equiv x \simeq_{\mathcal{A}} z$, *for every* $z : A$.

PROOF. By definition

$$\mathtt{idtoEqv}_{\mathcal{A}}(x, x, \mathtt{refl}_x) \equiv (\mathtt{refl}_x)^{P_x}_*(\mathtt{eqv}_x) \equiv \mathrm{id}_{P_x(x)}(\mathtt{eqv}_x) \equiv \mathtt{eqv}_x.$$

Using basic properties of transport we have that $\mathtt{idtoEqv}_{\mathcal{A}}(x, z, p * q) \equiv (p*q)^{P_x}_*(\mathtt{eqv}_x) = q^{P_x}_*(p^{P_x}_*(\mathtt{eqv}_x)) \equiv q^{P_x}_*(\mathtt{idtoEqv}_{\mathcal{A}}(x, y, p))$, where $p : x =_A y$ and $q : y =_A z$. Using path induction one shows that

$$C(x, y, p) \equiv q^{P_x}_*(\mathtt{idtoEqv}_{\mathcal{A}}(x, y, p)) \cong_{\mathcal{A}} p^{P_x}_*(\mathtt{eqv}_x) *_{\mathcal{A}} q^{P_y}_*$$

is inhabited, since $C(x, x, \mathtt{refl}_x) \equiv q^{P_x}_*(\mathtt{eqv}_x) \cong_{\mathcal{A}} \mathtt{eqv}_x *_{\mathcal{A}} q^{P_x}_*(\mathtt{eqv}_x)$ is inhabited by Typ_1. If $x, y : A$, we define $D : \prod_{p,q:x=_A y} \prod_{r:p=_{x=_A y} q} \mathcal{U}$ by

$$D(p, q, r) \equiv \mathtt{idtoEqv}_{\mathcal{A}}(x, y, p) \cong_{\mathcal{A}} \mathtt{idtoEqv}_{\mathcal{A}}(x, y, q),$$

for every $p, q : x =_A y$ and $r : p =_{x=_A y} q$. Since $\cong_{\mathcal{A}}$ (x, y) is an equivalence relation on $x =_A y$, there is some

$$i : D(p, p, \mathtt{refl}_p) \equiv \mathtt{idtoEqv}_{\mathcal{A}}(x, y, p) \cong_{\mathcal{A}} \mathtt{idtoEqv}_{\mathcal{A}}(x, y, p),$$

hence by path induction there is a dependent function

$$F_{xy} : \prod_{p,q:x=_A y} \prod_{r:p=_{x=_A y} q} D(p, q, r),$$

and we define

$$\mathtt{idtoEqv}^2_{\mathcal{A}} \equiv \lambda(x, y : A).F_{xy}.$$

□

If we consider A with the equality typoid structure as a codomain of id_A, then by Lemma 2.11.2 in [**16**]

$$\mathtt{idtoEqv}_{\mathcal{A}_0}(x, y, p) \equiv p^{P_x}_*(\mathtt{refl}_x) \equiv p^{z \mapsto x=_A z}_*(\mathtt{refl}_x) = \mathtt{refl}_x * p = p$$

i.e., $\mathtt{idtoEqv}_{\mathcal{A}_0}(x, y, p)$ is pointwisely equal to $\mathrm{id}_{x=_A y}$. In [**16**], section 2.10, the function $\mathtt{idtoEqv} : A =_{\mathcal{U}} B \to A \simeq_{\mathcal{U}} B$ is defined by

$$\mathtt{idtoEqv}(p) \equiv p^{\mathrm{id}_{\mathcal{U}}}_*,$$

for every $p : A =_{\mathcal{U}} B$. Since

$$\mathtt{idtoEqv}(\mathtt{refl}_A) \equiv (\mathtt{refl}_A)^{\mathrm{id}_{\mathcal{U}}}_* \equiv \mathrm{id}_{\mathrm{id}_{\mathcal{U}}(A)} \equiv \mathrm{id}_A,$$

$$\mathtt{idtoEqv}_{\mathrm{Uni}}(A, A, \mathtt{refl}_A) \equiv (\mathtt{refl}_A)^{P_A}_*(\mathrm{id}_A) \equiv \mathrm{id}_{P_A(A)}(\mathrm{id}_A) \equiv \mathrm{id}_A$$

the two functions agree on $\mathtt{refl}_A$, hence by path induction they are pointwisely equal. The definition of $\mathtt{idtoEqv}_{\mathcal{A}}$ is a common formulation of the functions `happly`, related to the axiom of function extensionality, and `idtoEqv`, related to the univalence axiom.

PROPOSITION 3.2.11. *If $(A, \simeq_{\mathcal{A}}), (B, \simeq_{\mathcal{B}})$ are pretypoids, and*

$$(x, y) \simeq_{\mathcal{A}\times\mathcal{B}} (x', y') \equiv (x \simeq_{\mathcal{A}} x') \times (y \simeq_{\mathcal{B}} y'),$$

there are dependent functions

$$T : \prod_{z,w:A\times B} \left((\mathtt{pr}_1(z) \simeq_{\mathcal{A}} \mathtt{pr}_1(w)) \times (\mathtt{pr}_2(z) \simeq_{\mathcal{B}} \mathtt{pr}_2(w)) \to z \simeq_{\mathcal{A}\times\mathcal{B}} w \right),$$

$$\Upsilon : \prod_{z,w:A\times B} \left(z \simeq_{\mathcal{A}\times\mathcal{B}} w \to (\mathtt{pr}_1(z) \simeq_{\mathcal{A}} \mathtt{pr}_1(w)) \times (\mathtt{pr}_2(z) \simeq_{\mathcal{B}} \mathtt{pr}_2(w)) \right)$$

such that for each $i \in \{1, 2\}$, where $C_1 \equiv A$ and $C_2 \equiv B$,

$$\prod_{z,w:A\times B} \mathtt{pr}_i \Big(\Upsilon(z, w, T(z, w, e_1, e_2)) \Big) \cong_{C_i} e_i.$$

PROOF. If $x : A$ and $y : B$, we find $G(x, y) : \prod_{w:A\times B} P(w)$, where

$$P(w) \equiv (x, y) \simeq_{\mathcal{A}\times\mathcal{B}} w \to (x \simeq_{\mathcal{A}} \mathtt{pr}_1(w)) \times (y \simeq_{\mathcal{B}} \mathtt{pr}_2(w)),$$

hence by the induction principle of the product type we get Υ. We define $H : \prod_{x':A} \prod_{y':B} P((x', y'))$ by $H(x', y') \equiv \mathrm{id}_{(x,y)\simeq_{\mathcal{A}\times\mathcal{B}}(x',y')}$, and then $H(x', y') : (x, y) \simeq_{\mathcal{A}\times\mathcal{B}} (x', y') \leftrightarrow (x \simeq_{\mathcal{A}} x') \times (y \simeq_{\mathcal{B}} y')$. Hence $G(x, y)$ is given again by the induction principle of the product type. For T and the last two dependent equivalences we work similarly. □

COROLLARY 3.2.12. *If $(A, \simeq_{\mathcal{A}}), (B, \simeq_{\mathcal{B}})$ are pretypoids, then $\mathtt{pr}_1, \mathtt{pr}_2$ are pretypoid functions.*

PROOF. We define

$$\Phi_{\mathtt{pr}_1} : \prod_{z,w:A\times B} \prod_{e:z\simeq_{A\times} w} \mathtt{pr}_1(z) \simeq_{\mathcal{A}} \mathtt{pr}_1(w),$$

$$\Phi_{\mathtt{pr}_1}(z, w, e) \equiv \mathtt{pr}_1(\Upsilon(z, w, e)).$$

We define $\Phi_{\mathtt{pr}_2}(z, w, e) \equiv \mathtt{pr}_2(\Upsilon(z, w, e))$ similarly. □

Next we use the notations $e_1 \equiv \Phi_{\mathtt{pr}_1}(z, w, e)$ and $e_2 \equiv \Phi_{\mathtt{pr}_2}(z, w, e)$.

PROPOSITION 3.2.13. *If $\mathcal{A}, \mathcal{B}$ are typoids, the structure*

$$\mathcal{A} \times \mathcal{B} \equiv (A \times B, \simeq_{\mathcal{A}\times\mathcal{B}}, \mathtt{eqv}_{\mathcal{A}\times\mathcal{B}}, *_{\mathcal{A}\times\mathcal{B}}, {}^{-1_{\mathcal{A}\times\mathcal{B}}}, \cong_{\mathcal{A}\times\mathcal{B}})$$

is a typoid, where for every $z, w, u : A\times B$ and $e, e' : z =_{A\times B} w, d : w =_{A\times B} u$

$$\mathtt{eqv}_z \equiv T(z, z, \mathtt{eqv}_{\mathtt{pr}_1(z)}, \mathtt{eqv}_{\mathtt{pr}_2(z)}),$$

$$e *_{\mathcal{A}\times\mathcal{B}} d \equiv T(z, u, e_1 *_{\mathcal{A}} d_1, e_2 *_{\mathcal{B}} d_2),$$

$$e^{-1_{\mathcal{A}\times\mathcal{B}}} \equiv T(w, z, e_1^{-1_{\mathcal{A}}}, e_2^{-1_{\mathcal{B}}}),$$
$$e \cong_{\mathcal{A}\times\mathcal{B}} e' \equiv (e_1 \cong_{\mathcal{A}} {e_1}') \times (e_2 \cong_{\mathcal{B}} {e_2}').$$

PROOF. Left to the reader. □

COROLLARY 3.2.14. *If $\mathcal{A}, \mathcal{B}$ are typoids, $\mathtt{pr}_1, \mathtt{pr}_2$ are typoid functions.*

PROOF. We show only that $\mathtt{pr}_1$ is a typoid function. The equivalence $\Phi_{\mathtt{pr}_1}(z, z, \mathtt{eqv}_z) \cong_{\mathcal{A}} \mathtt{eqv}_{\mathtt{pr}_1(z)}$ follows from the equivalence

$$\begin{aligned}(\mathtt{eqv}_z)_1 &\equiv \Phi_{\mathtt{pr}_1}(z, z, \mathtt{eqv}_z)\\ &\equiv \mathtt{pr}_1(\Upsilon(z, z, \mathtt{eqv}_z))\\ &\equiv \mathtt{pr}_1(\Upsilon(z, z, T(z, z, \mathtt{eqv}_{\mathtt{pr}_1(z)}, \mathtt{eqv}_{\mathtt{pr}_2(z)})))\\ &\cong_{\mathcal{A}} \mathtt{eqv}_{\mathtt{pr}_1(z)}.\end{aligned}$$

We rewrite the equivalence

$$\Phi_{\mathtt{pr}_1}(z, u, e *_{\mathcal{A}\times\mathcal{B}} d) \cong_{\mathcal{A}} \Phi_{\mathtt{pr}_1}(z, w, e) *_{\mathcal{A}} \Phi_{\mathtt{pr}_1}(w, u, d)$$

as $(e *_{\mathcal{A}\times\mathcal{B}} d)_1 \cong_{\mathcal{A}} (e_1 *_{\mathcal{A}} d_1)$, which follows by the definition of $e *_{\mathcal{A}\times\mathcal{B}} d$ and the relation between Υ and T in Proposition 3.2.11. Since $e_1 \equiv \Phi_{\mathtt{pr}_1}(z, w, e)$ and $d_1 \equiv \Phi_{\mathtt{pr}_1}(z, w, d)$, by the definition of $e \cong_{\mathcal{A}\times\mathcal{B}} d$ we get the following 2-associate of $\mathtt{pr}_1$ with respect to $\Phi_{\mathtt{pr}_1}$

$$\Phi^2_{\mathtt{pr}_1} : \prod_{z,w:A\times B} \prod_{e,d:z\simeq_{\mathcal{A}\times\mathcal{B}} w} \prod_{i:e\cong_{\mathcal{A}\times\mathcal{B}} d} \Phi_{\mathtt{pr}_1}(z, w, e) \cong_{\mathcal{A}} \Phi_{\mathtt{pr}_1}(z, w, d)$$

$$\Phi^2_{\mathtt{pr}_1}(z, w, e, d, i) \equiv \mathtt{pr}_1(i).$$

□

COROLLARY 3.2.15. *If $\mathcal{A}, \mathcal{B}$ are typoids, $z, w : A \times B$, and $e : z \simeq_{\mathcal{A}\times\mathcal{B}} w$,*

$$T(z, w, e_1, e_2) \cong_{\mathcal{A}\times\mathcal{B}} e.$$

PROOF. If $\tau \equiv T(z, w, e_1, e_2)$, then $\tau \cong_{\mathcal{A}\times\mathcal{B}} e \equiv (\tau_1 \cong_{\mathcal{A}} e_1) \times (\tau_2 \cong_{\mathcal{A}} e_2)$. By Proposition 3.2.11 we have that $\tau_1 \equiv \mathtt{pr}_1(\Upsilon(z, w, T(z, w, e_1, e_2))) \cong_{\mathcal{A}} e_1$, and similarly $\tau_2 \cong_{\mathcal{B}} e_2$. □

COROLLARY 3.2.16. *If $\mathcal{A}, \mathcal{B}$ are typoids, $z, w : A \times B$, and $e_1, d_1 : \mathtt{pr}_1(z) \simeq_{\mathcal{A}} \mathtt{pr}_1(w)$, $e_2, d_2 : \mathtt{pr}_2(z) \simeq_{\mathcal{A}} \mathtt{pr}_2(w)$ such that $e_1 \cong_{\mathcal{A}} d_1$ and $e_2 \cong_{\mathcal{A}} d_2$, then*

$$T(z, w, e_1, e_2) \cong_{\mathcal{A}\times\mathcal{B}} T(z, w, d_1, d_2).$$

PROOF. If $\tau \equiv T(z,w,e_1,e_2)$ and $\tau' \equiv T(z,w,d_1,d_2)$, then by Proposition 3.2.11 we have that $\tau_1 \equiv \mathtt{pr}_1(\Upsilon(z,w,T(z,w,e_1,e_2)) \cong_{\mathcal{A}} e_1$ and $\tau_2 \equiv \mathtt{pr}_2(\Upsilon(z,w,T(z,w,e_1,e_2)) \cong_{\mathcal{B}} e_2$. In a similar manner we have that ${\tau_1}' \equiv \mathtt{pr}_1(\Upsilon(z,w,T(z,w,d_1,d_2)) \cong_{\mathcal{A}} d_1$ and ${\tau_2}' \equiv \mathtt{pr}_2(\Upsilon(z,w,T(z,w,d_1,d_2)) \cong_{\mathcal{B}} d_2$. By our hypothesis and the definition of $\cong_{\mathcal{A}\times\mathcal{B}}$ we get $\tau \cong_{\mathcal{A}\times\mathcal{B}} \tau'$. □

If $\mathcal{A},\mathcal{B}$ are typoids and $f : A \to B$, the type "f is a typoid function" is

$$\mathtt{Typfun}(f) \equiv \sum_{\Phi_f:\prod_{x,y:A}\prod_{e:x\simeq_{\mathcal{A}}y} f(x)\simeq_{\mathcal{B}}f(y)} \Bigg[\Bigg(\prod_{x,y:A}\prod_{e:x\simeq_{\mathcal{A}}y}\prod_{d:y\simeq_{\mathcal{A}}z} \Big(\Phi_f(x,x,\mathtt{eqv}_x) \cong_{\mathcal{B}} \mathtt{eqv}_{f(x)}\Big)\times \Big(\Phi_f(x,z,e *_{\mathcal{A}} d) \cong_{\mathcal{B}} \Phi_f(x,y,e) *_{\mathcal{B}} \Phi_f(y,z,d)\Big)\Bigg)\times \times\Bigg(\prod_{x,y:A}\prod_{e,d:x\simeq_{\mathcal{A}}y}\prod_{i:e\cong_{\mathcal{A}}d} \Phi_f(x,y,e) \cong_{\mathcal{B}} \Phi_f(x,y,d)\Bigg)\Bigg].$$

A canonical element of $\mathtt{Typfun}(f)$ is a pair $(\Phi_f,(U,\Phi_f^2))$, or for simplicity a triplet (Φ_f,U,Φ_f^2), where

$$U : \prod_{x,y:A}\prod_{e:x\simeq_{\mathcal{A}}y}\prod_{d:y\simeq_{\mathcal{A}}z} \Big(\Phi_f(x,x,\mathtt{eqv}_x) \cong_{\mathcal{B}} \mathtt{eqv}_{f(x)}\Big)\times \Big(\Phi_f(x,z,e *_{\mathcal{A}} d) \cong_{\mathcal{B}} \Phi_f(x,y,e) *_{\mathcal{B}} \Phi_f(y,z,d)\Big)$$

and

$$\Phi_f^2 : \prod_{x,y:A}\prod_{e,d:x\simeq_{\mathcal{A}}y}\prod_{i:e\cong_{\mathcal{A}}d} \Phi_f(x,y,e) \cong_{\mathcal{B}} \Phi_f(x,y,d).$$

DEFINITION 3.2.17. The *exp-typoid* B^A of A,B is defined as

$$B^A \equiv \sum_{f:A\to B} \mathtt{Typfun}(f).$$

If $\phi \equiv (f,\Phi_f,U,\Phi_f^2)$ and $\theta \equiv (g,\Phi_g,W,\Phi_g^2)$ are two canonical elements of B^A, we define

$$\phi \simeq_{B^A} \theta \equiv \sum_{\Theta_{f,g}:\prod_{x:A} f(x)\simeq_{\mathcal{B}}g(x)} \Bigg(\prod_{x,y:A}\prod_{e:x\simeq_{\mathcal{A}}y} \Phi_{f(x,y,e)} *_{\mathcal{B}} \Theta_{f,g}(y) \cong_{\mathcal{B}}$$

$$\Theta_{f,g}(x) *_{\mathcal{B}} \Phi_g(x,y,e)\Big).$$

A canonical element e of $\phi \simeq_{B^A} \theta$ is a pair $(\Theta_{f,g}, \Theta^2_{f,g})$, where

$$\Theta^2_{f,g} : \prod_{x,y:A} \prod_{e:x\simeq_{\mathcal{A}} y} \Phi_{f(x,y,e)} *_{\mathcal{B}} \Theta_{f,g}(y) \cong_{\mathcal{B}} \Theta_{f,g}(x) *_{\mathcal{B}} \Phi_g(x,y,e).$$

If ϕ is a canonical element of B^A we define $\mathtt{eqv}_\phi : \phi \simeq_{B^A} \phi$ as the pair $(\Theta_{f,f}, \Theta^2_{f,f})$, where $\Theta_{f,f} \equiv \lambda(x:A).\mathtt{eqv}_{f(x)} : \prod_{x:A} f(x) \simeq_{\mathcal{B}} f(x)$ and $\Theta^2_{f,f}(x,y,e)$ is the obvious proof that the corresponding diagram of equivalences commutes. If $\phi \equiv (f, \Phi_f, U, \Phi^2_f), \theta \equiv (g, \Phi_g, W, \Phi^2_g), \eta \equiv (h, \Phi_h, V, \Phi^2_h)$ are canonical elements of B^A and $e \equiv (\Theta_{f,g}, \Theta^2_{f,g}) : \phi \simeq_{B^A} \theta$ and $d \equiv (\Theta_{g,h}, \Theta^2_{g,h}) : \theta \simeq_{B^A} \eta$, we define

$$e *_{B^A} d \equiv (\Theta_{f,h}, \Theta^2_{f,h}) : \phi \simeq_{B^A} \eta$$
$$\Theta_{f,h} \equiv \lambda(x:A).\Theta_{f,g}(x) *_{\mathcal{B}} \Theta_{g,h}(x).$$

A term $\Theta^2_{f,h}(x,y,e)$ of type

$$\Phi_{f(x,y,e)} *_{\mathcal{B}} \Theta_{f,h}(y) \cong_{\mathcal{B}} \Theta_{f,h}(x) *_{\mathcal{B}} \Phi_h(x,y,e) \equiv$$
$$\Phi_{f(x,y,e)} *_{\mathcal{B}} (\Theta_{f,g}(y) *_{\mathcal{B}} \Theta_{g,h}(y)) \cong_{\mathcal{B}} (\Theta_{f,h}(x) *_{\mathcal{B}} \Theta_{g,h}(x)) *_{\mathcal{B}} \Phi_h(x,y,e)$$

is easily found.

If $e \equiv (\Theta_{f,g}, \Theta^2_{f,g}) : \phi \simeq_{B^A} \theta$, we define

$$e^{-1_{B^A}} \equiv (\Theta^{-1}_{f,g}, [\Theta^2_{f,g}]^{-1}) : \theta \simeq_{B^A} \phi,$$

where $\Theta^{-1}_{f,g} : \prod_{x:A} g(x) \simeq_{\mathcal{B}} f(x)$ is defined by

$$\Theta^{-1}_{f,g}(x) \equiv [\Theta_{f,g}(x)]^{-1_{\mathcal{B}}},$$

for every $x : A$, and $[\Theta^2_{f,g}]^{-1}(y,x,e)$ is of type $\Phi_g(y,x,e) *_{\mathcal{B}} \Theta_{f,g}(x)^{-1} \cong_{\mathcal{B}} \Theta_{f,g}(y)^{-1} *_{\mathcal{B}} \Phi_f(y,x,e)$ i.e., a proof of the commutativity of the obvious diagram of equivalences.

3.3. Univalent typoids

Definition 3.3.1. A typoid $\mathcal{A}$ is called *univalent*, if there are

$$\mathtt{Ua}_{\mathcal{A}} : \prod_{x,y:A} \prod_{e:x\simeq_{\mathcal{A}} y} x =_A y,$$

$$\mathtt{Ua}^2_{\mathcal{A}} : \prod_{x,y:A} \prod_{e,d:x\simeq_{\mathcal{A}} y} \prod_{i:e\cong_{\mathcal{A}} d} \mathtt{Ua}_{\mathcal{A}}(x,y,e) = \mathtt{Ua}_{\mathcal{A}}(x,y,d)$$

such that for every $x, y : A, p : x =_A y$ and $e : x \simeq_{\mathcal{A}} y$ we have that

$$\mathtt{Ua}_{\mathcal{A}}(x, y, \mathtt{IdtoEqv}_{\mathcal{A}}(x, y, p)) = p,$$

$$\mathtt{IdtoEqv}_{\mathcal{A}}(x, y, \mathtt{Ua}_{\mathcal{A}}(x, y, e)) \cong_{\mathcal{A}} e,$$

where $\mathtt{IdtoEqv}_{\mathcal{A}}$ is an 1-associate of id_A (from $\mathcal{A}_0$ to $\mathcal{A}$) with respect to which id_A is strict[1]. We call a univalent typoid *strictly* univalent, if $\mathtt{Ua}_{\mathcal{A}}(x, x, \mathtt{eqv}_x) \equiv \mathtt{refl}_x$.

The equality typoid $\mathcal{A}_0$ is strictly univalent, if we consider

$$\mathtt{IdtoEqv}_{\mathcal{A}_0}(x, y, p) \equiv p \equiv \mathtt{Ua}_{\mathcal{A}_0}(x, y, p),$$

for every $x, y : A$ and $p : x \simeq_{\mathcal{A}_0} y$. The function extensionality axiom guarantees that the typoid of functions $\mathrm{Fun}(A, B)$ is univalent, and Voevodsky's univalence axiom that the universal typoid Uni is univalent. We need only to explain why the functions `funext` and `ua` satisfy the conditions: if $H, H' : f \simeq_{A\to B} g$ such that $H \cong_{A\to B} H'$, then $\mathtt{funext}(H) = \mathtt{funext}(H')$, and if $(f, u), (g, w) : A \simeq_{\mathcal{U}} B$ such that $(f, u) \cong_{\mathcal{U}} (g, w)$, then $\mathtt{ua}((f, u)) = \mathtt{ua}((g, w))$, respectively. By the function extensionality axiom if $H, H' : f \simeq_{A\to B} g$, then there is $p : H = H'$, hence the application function of `funext` satisfies $\mathtt{ap}_{\mathtt{funext}}(p) : \mathtt{funext}(H) = \mathtt{funext}(H')$. By Theorem 2.7.2 of [**16**] we have that

$$\big((f, u) =_{A\simeq_{\mathrm{Uni}} B} (g, w)\big) \simeq_{\mathcal{U}} \sum_{p:f=g} \left(p_*^{f\mapsto \mathtt{isequiv}(f)}(u) = w\right).$$

By the function extensionality axiom the hypothesis $(f, u) \cong_{\mathrm{Uni}} (g, w)$ implies that $f = g$, while a term of type $p_*^{f\mapsto \mathtt{isequiv}(f)}(u) = w$ is found by the equality of all terms of type $\mathtt{isequiv}(g)$. Hence the hypothesis $(f, u) \cong_{\mathrm{Uni}} (g, w)$ implies $(f, u) =_{A\simeq_U B} (g, w)$ and we use the application function of `ua` to get a term of type $\mathtt{ua}((f, u)) = \mathtt{ua}((g, w))$.

The next proposition is a common reformulation of properties of the functions `funext` and `ua` found in sections 2.9 and 2.10 of [**16**], respectively.

Proposition 3.3.2. *If $\mathcal{A}$ is a univalent typoid, the identity function $\mathrm{id}_A : A \to A$ is a typoid function from $\mathcal{A}$ to $\mathcal{A}_0$, with $\mathtt{Ua}^2_{\mathcal{A}}$ as a 2-associate of id_A with respect to its 1-associate $\mathtt{Ua}_{\mathcal{A}}$.*

[1]In Proposition 3.2.10 we showed the existence of such an associate of id_A but in general we don't need to use the specific definition of $\mathtt{idtoEqv}_{\mathcal{A}}$. This is crucial in proving that the product of univalent typoids is a univalent typoid. For this reason we use a different notation for this abstract 1-associate of id_A. Of course, by path induction $\mathtt{idtoEqv}_{\mathcal{A}}$ and $\mathtt{IdtoEqv}_{\mathcal{A}}$ are pointwisely equal, hence by function extensionality they are equal.

PROOF. We show that $\Phi_{\mathrm{id}_A} \equiv \mathtt{Ua}_{\mathcal{A}}$ and $\Phi^2_{\mathrm{id}_A} \equiv \mathtt{Ua}^2_{\mathcal{A}}$ are associates of id_A. By definition $\mathtt{Ua}_{\mathcal{A}}(x,x,\mathtt{eqv}_x) \equiv \mathtt{Ua}_{\mathcal{A}}(x,x,\mathtt{IdtoEqv}_{\mathcal{A}}(x,x,\mathtt{refl}_x)) = \mathtt{refl}_x$. Since by definition we have that $\mathtt{IdtoEqv}_{\mathcal{A}}(x,y,\mathtt{Ua}_{\mathcal{A}}(x,y,e_1)) \cong_{\mathcal{A}} e_1$ and $\mathtt{IdtoEqv}_{\mathcal{A}}(y,z,\mathtt{Ua}_{\mathcal{A}}(y,z,e_2)) \cong_{\mathcal{A}} e_2$, by Typ_4 we get

$$\begin{aligned} e_1 *_{\mathcal{A}} e_2 &\equiv_{\mathcal{A}} \mathtt{IdtoEqv}_{\mathcal{A}}(x,y,\mathtt{Ua}_{\mathcal{A}}(x,y,e_1)) *_{\mathcal{A}} \mathtt{IdtoEqv}_{\mathcal{A}}(y,z,\mathtt{Ua}_{\mathcal{A}}(y,z,e_2)) \\ &\equiv_{\mathcal{A}} \mathtt{IdtoEqv}_{\mathcal{A}}(x,z,[\mathtt{Ua}_{\mathcal{A}}(x,y,e_1) *_{\mathcal{A}} \mathtt{Ua}_{\mathcal{A}}(y,z,e_2)])) \end{aligned}$$

If $B \equiv \mathtt{Ua}_{\mathcal{A}}\big(x,z,\mathtt{IdtoEqv}_{\mathcal{A}}(x,z,[\mathtt{Ua}_{\mathcal{A}}(x,y,e_1) *_{\mathcal{A}} \mathtt{Ua}_{\mathcal{A}}(y,z,e_2)])\big)\big)$, by the existence of $\mathtt{Ua}^2_{\mathcal{A}}$ we get a term of type $\mathtt{Ua}_{\mathcal{A}}(x,z,e_1 *_{\mathcal{A}} e_2) = B$, hence a term of type $\mathtt{Ua}_{\mathcal{A}}(x,z,e_1 *_{\mathcal{A}} e_2) = \mathtt{Ua}_{\mathcal{A}}(x,y,e_1) *_{\mathcal{A}} \mathtt{Ua}_{\mathcal{A}}(y,z,e_2)$. □

THEOREM 3.3.3. *Let $\mathcal{A},\mathcal{B}$ be typoids and $f : A \to B$.*
(i) If $\mathcal{A}$ is univalent, then f is a typoid function.
(ii) If $\mathcal{A}$ is strictly univalent, then f is a strict typoid function with respect to its 1-associate given in the proof of (i).

PROOF. (i) Through the correspondences

$$x \simeq_{\mathcal{A}} y \overset{\mathtt{Ua}_{\mathcal{A}}(x,y)}{\longrightarrow} x =_A y \overset{\mathtt{ap}_f(x,y)}{\longrightarrow} f(x) =_B f(y) \overset{\mathtt{IdtoEqv}_{\mathcal{B}}(f(x),f(y))}{\longrightarrow} f(x) \simeq_{\mathcal{B}} f(y)$$

we define the dependent function $\Phi_f : \prod_{x,y:A} \prod_{e:x\simeq_{\mathcal{A}} y} f(x) \simeq_{\mathcal{B}} f(y)$ by

$$\Phi_f(x,y,e) \equiv \mathtt{IdtoEqv}_{\mathcal{B}}\Big(f(x),f(y),\mathtt{ap}_f(x,y,\mathtt{Ua}_{\mathcal{A}}(x,y,e))\Big).$$

By the proof of Proposition 3.3.2 there is $r : \mathtt{Ua}_{\mathcal{A}}(x,x,\mathtt{eqv}_x) = \mathtt{refl}_x$, and by $\mathtt{ap}^2_f$ we get a term $r' : \mathtt{ap}_f(x,x,\mathtt{Ua}_{\mathcal{A}}(x,x,\mathtt{eqv}_x)) = \mathtt{ap}_f(x,x,\mathtt{refl}_x) \equiv \mathtt{refl}_{f(x)}$. Since $\mathtt{IdtoEqv}^2_{\mathcal{B}}$ is of type

$$\prod_{x',y':B} \prod_{p',q':x'=_B y'} \prod_{r':p'=q'} \mathtt{IdtoEqv}_{\mathcal{B}}(x',y',p') \cong_{\mathcal{B}} \mathtt{IdtoEqv}_{\mathcal{B}}(x',y',q'),$$

$\mathtt{IdtoEqv}^2_{\mathcal{B}}(f(x),f(x),\mathtt{ap}_f(x,x,\mathtt{Ua}_{\mathcal{A}}(x,x,\mathtt{eqv}_x)),\mathtt{refl}_{f(x)},r')$ is of type

$$\mathtt{IdtoEqv}_{\mathcal{B}}(f(x),f(x),\mathtt{ap}_f(x,x,\mathtt{Ua}_{\mathcal{A}}(x,x,\mathtt{eqv}_x))) \cong_{\mathcal{B}}$$
$$\mathtt{IdtoEqv}_{\mathcal{B}}(f(x),f(x),\mathtt{refl}_{f(x)}).$$

By the definition of Φ_f and the fact that $\mathtt{IdtoEqv}_{\mathcal{B}}(f(x),f(x),\mathtt{refl}_{f(x)}) \equiv \mathtt{eqv}_{f(x)}$ we get $\Phi_f(x,x,\mathtt{eqv}_x) \cong_{\mathcal{B}} \mathtt{eqv}_{f(x)}$. By the existence of $\mathtt{ap}^2_f$ and $\mathtt{IdtoEqv}^2_{\mathcal{B}}$ we get

$$\begin{aligned} \Phi_f(x,z,e *_{\mathcal{A}} d) &\equiv \mathtt{IdtoEqv}_{\mathcal{B}}\Big(f(x),f(z),\mathtt{ap}_f(x,z,\mathtt{Ua}_{\mathcal{A}}(x,z,e *_{\mathcal{A}} d))\Big) \\ &\cong_{\mathcal{B}} \mathtt{IdtoEqv}_{\mathcal{B}}\Big(f(x),f(z),\mathtt{ap}_f(x,z,\mathtt{Ua}_{\mathcal{A}}(x,y,e)* \end{aligned}$$

$$\begin{aligned}
&\qquad\qquad \mathtt{Ua}_{\mathcal{A}}(y,z,d))\Big)\\
&\cong_{\mathcal{B}} \mathtt{IdtoEqv}_{\mathcal{B}}\Big(f(x),f(z),\mathtt{ap}_f(x,y,\mathtt{Ua}_{\mathcal{A}}(x,y,e))*_{\mathcal{B}}\\
&\qquad\qquad \mathtt{ap}_f(y,z,\mathtt{Ua}_{\mathcal{A}}(y,z,d))\Big)\\
&\cong_{\mathcal{B}} \mathtt{IdtoEqv}_{\mathcal{B}}\Big(f(x),f(y),\mathtt{ap}_f(x,y,\mathtt{Ua}_{\mathcal{A}}(x,y,e))\Big)*_{\mathcal{B}}\\
&\qquad \mathtt{IdtoEqv}_{\mathcal{B}}\Big(f(y),f(z),\mathtt{ap}_f(y,z,\mathtt{Ua}_{\mathcal{A}}(y,z,d))\Big)\\
&\equiv \Phi_f(x,y,e)*_{\mathcal{B}}\Phi_f(y,z,d).
\end{aligned}$$

We define $\Phi^2_f : \prod_{x,y:A}\prod_{e,d:x\simeq_{\mathcal{A}}y}\prod_{i:e\cong_{\mathcal{A}}d}\Phi_f(x,y,e)\cong_{\mathcal{B}}\Phi_f(x,y,d)$ by

$$\begin{aligned}
\Phi^2_f(x,y,e,d,i) \equiv \mathtt{IdtoEqv}^2_{\mathcal{B}}\Bigg(&f(x),f(y),\mathtt{ap}_f(x,y,\mathtt{Ua}_{\mathcal{A}}(x,y,e)),\\
&\mathtt{ap}_f(x,y,\mathtt{Ua}_{\mathcal{A}}(x,y,d)),\\
&\mathtt{ap}^2_f\Big(x,y,\mathtt{ap}_f(x,y,\mathtt{Ua}_{\mathcal{A}}(x,y,e)),\mathtt{ap}_f(x,y,\mathtt{Ua}_{\mathcal{A}}(x,y,d)),\mathtt{Ua}^2_{\mathcal{A}}(x,y,e,d,i)\Big)\Bigg).
\end{aligned}$$

Since the term $\mathtt{Ua}^2_{\mathcal{A}}(x,y,e,d,i)$ is of type $\mathtt{Ua}_{\mathcal{A}}(x,y,e) = \mathtt{Ua}_{\mathcal{A}}(x,y,d)$ and the terms $\mathtt{ap}_f(x,y,\mathtt{Ua}_{\mathcal{A}}(x,y,e))$ and $\mathtt{ap}_f(x,y,\mathtt{Ua}_{\mathcal{A}}(x,y,d))$ are of type $f(x) =_B f(y)$, the term

$$\mathtt{ap}^2_f\Big(x,y,\mathtt{ap}_f(x,y,\mathtt{Ua}_{\mathcal{A}}(x,y,e)),\mathtt{ap}_f(x,y,\mathtt{Ua}_{\mathcal{A}}(x,y,d)),\mathtt{Ua}^2_{\mathcal{A}}(x,y,e,d,i)\Big)$$

is of type $\mathtt{ap}_f(x,y,\mathtt{Ua}_{\mathcal{A}}(x,y,e)) = \mathtt{ap}_f(x,y,\mathtt{Ua}_{\mathcal{A}}(x,y,d))$. Hence by the type of $\mathtt{IdtoEqv}^2_{\mathcal{B}}$ and the definition of Φ_f we get that $\Phi^2_f(x,y,e,d,i)$ is of type $\Phi_f(x,y,e)\cong_{\mathcal{B}}\Phi_f(x,y,d)$.

(ii) By the proof of Proposition 3.2.10 we have that

$$\begin{aligned}
\Phi_f(x,x,\mathtt{eqv}_x) &\equiv \mathtt{IdtoEqv}_{\mathcal{B}}\Big(f(x),f(x),\mathtt{ap}_f(x,x,\mathtt{Ua}_{\mathcal{A}}(x,x,\mathtt{eqv}_x))\Big)\\
&\equiv \mathtt{IdtoEqv}_{\mathcal{B}}\Big(f(x),f(x),\mathtt{ap}_f(x,x,\mathtt{refl}_x))\Big)\\
&\equiv \mathtt{IdtoEqv}_{\mathcal{B}}(f(x),f(x),\mathtt{refl}_{f(x)}))\\
&\equiv \mathtt{eqv}_{f(x)}.
\end{aligned}$$

□

THEOREM 3.3.4. *If $\mathcal{A},\mathcal{B}$ are univalent typoids, then $\mathcal{A}\times\mathcal{B}$ is a univalent typoid.*

PROPOSITION 3.3.5. *If $\mathcal{A}, \mathcal{B}$ are typoids and $\mathcal{A} \times \mathcal{B}$ is univalent, then $\mathcal{A}, \mathcal{B}$ are univalent.*

3.4. "Higher" typoids

If A is a type, *a* (mere) propositional truncation of A is a type B such that $\mathtt{isProp}(B)$, if A is inhabited, then B is inhabited, and if $f : A \to C$, where $\mathtt{isProp}(C)$, there is $f_B : B \to C$. It is easy to see that if there is a type B' satisfying the above properties, then B, B' are logically equivalent, hence equivalent in $\mathcal{U}$, and by univalence equal in $\mathcal{U}$. In [**16**], section 3.7, the notion of *the* propositional truncation of a type A is implemented through the definition of the higher inductive type $||A||$.

In the setting of typoids we interpret the notion of the propositional truncation as the truncated typoid $\mathcal{A}^t$. Starting from a typoid structure on a type A we define a new typoid stucture on A, which behaves accordingly. Hence, we keep the same type and we change the typoid structure, while in the approach within types the type is changed.

DEFINITION 3.4.1. If $A : \mathcal{U}$, we call the typoid

$$\mathcal{A}^t \equiv (A, \simeq_{\mathcal{A}^t}, \mathtt{eqv}_{\mathcal{A}^t}, *_{\mathcal{A}^t}, {}^{-1_{\mathcal{A}^t}}, \cong_{\mathcal{A}^t})$$

truncated, if for every $x, y, z : A$, $e, e' : x \simeq_{\mathcal{A}^t} y$, and $d : y \sim_{\mathcal{A}^t} z$ we define

$$x \simeq_{\mathcal{A}^t} y \equiv \mathbf{1},$$
$$\mathtt{eqv}_{\mathcal{A}^t}(x) \equiv 0_{\mathbf{1}},$$
$$*_{\mathcal{A}^t}(x, y, z, e, d) \equiv 0_{\mathbf{1}},$$
$${}^{-1_{\mathcal{A}^t}}(x, y, e) \equiv 0_{\mathbf{1}},$$
$$\cong_{\mathcal{A}^t}(x, y, e, e') \equiv (e = e').$$

The proof that $\mathcal{A}^t$ is a typoid is immediate. One needs only to take into account that $\mathtt{isProp}(\mathbf{1})$, hence $\mathtt{isSet}(\mathbf{1})$ (see Lemma 3.3.4 of [**16**]).

PROPOSITION 3.4.2. *If $A : \mathcal{U}$, $\mathcal{B}$ is a typoid and $f : B \to A$, then f is a typoid function from $\mathcal{B}$ to $\mathcal{A}^t$.*

PROOF. Let $x, y, z : B$, $e, e' : x \simeq_{\mathcal{B}} y$ $i : e \cong_{\mathcal{B}} e'$, and $d : y \simeq_{\mathcal{B}} z$ We define $\Phi_f(x, y, e) \equiv 0_{\mathbf{1}}$, hence $\Phi_f(x, y, e) : f(x) \simeq_{\mathcal{A}^t} f(y)$, and we also define $\Phi^2_f(x, y, e, e', i) \equiv \mathtt{refl}_{0_{\mathbf{1}}}$, hence $\Phi^2_f(x, y, e, e', i) : 0_{\mathbf{1}} =_{\mathbf{1}} 0_{\mathbf{1}}$. Clearly, $\Phi_f(x, x, \mathtt{eqv}_x) \equiv 0_{\mathbf{1}} \equiv e_{f(x)}$, and $\Phi_f(x, z, e_1 *_{\mathcal{B}} e_2) \equiv 0_{\mathbf{1}} = (0_{\mathbf{1}} *_{\mathcal{A}^t} 0_{\mathbf{1}}) \equiv \Phi_f(x, y, e_1) *_{\mathcal{A}^t} \Phi_f(y, z, e_2)$.

□

COROLLARY 3.4.3. *If $A, B : \mathcal{U}$ and $f : B \to A$, then f is a typoid function from $\mathcal{B}^t$ to $\mathcal{A}^t$.*

If $f : B \to A$, we also use the notation f^t for f to indicate that we view f as a typoid function from B^t to A^t.

PROPOSITION 3.4.4. *If $A : \mathcal{U}$ such that $\mathtt{isProp}(A)$, then $\mathcal{A}^t$ is univalent.*

PROOF. If $\Omega : \mathtt{isProp}(A) \equiv \prod_{x,y:A}(x =_A y)$, $d, e : \mathbf{1}$ and $i : d \cong_{\mathcal{A}^t} e \equiv (d = e)$, we define

$$\mathtt{Ua}_{\mathcal{A}^t}(x, y, d) \equiv \Omega(x, y),$$

$$\mathtt{Ua}^2_{\mathcal{A}^t}(x, y, d, e, i) \equiv \mathtt{refl}_{\Omega(x,y)},$$

hence $\mathtt{Ua}_{\mathcal{A}^t}(x, y, d) : x =_A y$ and $\mathtt{Ua}^2_{\mathcal{A}^t}(x, y, d, e, i) : \Omega(x, y) = \Omega(x, y) \equiv \mathtt{Ua}_{\mathcal{A}^t}(x, y, d) = \mathtt{Ua}_{\mathcal{A}^t}(x, y, e)$. Let $\mathtt{IdtoEqv}_{\mathcal{A}^t}$ be an 1-associate of id_A seen as function from $\mathcal{A}_0$ to $\mathcal{A}^t$. Since $\mathtt{IdtoEqv}_{\mathcal{A}^t}(x, y, \mathtt{Ua}_{\mathcal{A}^t}(x, y, d)) : \mathbf{1}$, and $d : \mathbf{1}$, we get

$$\mathtt{IdtoEqv}_{\mathcal{A}^t}(x, y, \mathtt{Ua}_{\mathcal{A}^t}(x, y, d)) = d,$$

i.e., $\mathtt{IdtoEqv}_{\mathcal{A}^t}(x, y, \mathtt{Ua}_{\mathcal{A}^t}(x, y, d)) \cong_{\mathcal{A}^t} d$. Since

$$\mathtt{Ua}_{\mathcal{A}^t}(x, y, \mathtt{IdtoEqv}_{\mathcal{A}^t}(x, y, p)) : x =_A y,$$

$p : x =_A y$ and $\mathtt{isProp}(A) \to \mathtt{isSet}(A)$, we get

$$\mathtt{Ua}_{\mathcal{A}^t}(x, y, \mathtt{IdtoEqv}_{\mathcal{A}^t}(x, y, p)) = p.$$

□

Using Theorem 3.3.3(i) we get the following corollary.

COROLLARY 3.4.5. *If $A : \mathcal{U}$ such that $\mathtt{isProp}(A)$, $\mathcal{B}$ is a typoid and $f : A \to B$, then f is a typoid function from $\mathcal{A}^t$ to $\mathcal{B}$.*

PROPOSITION 3.4.6. *If $A : \mathcal{U}$, $\mathcal{B}$ is a typoid such that $\mathtt{isProp}(B)$, and $f : A \to B$, then f is a typoid function from $\mathcal{A}^t$ to $\mathcal{B}$.*

PROOF. By Corollary 3.4.3 f is a typoid function from $\mathcal{A}^t$ to $\mathcal{B}^t$, while by Corollary 3.4.5 id_B is a typoid function from $\mathcal{B}^t$ to $\mathcal{B}$. By Proposition 3.2.9 $f \equiv \mathrm{id}_B \circ f$ is a typoid function from $\mathcal{A}^t$ to $\mathcal{B}$. □

Hence, in this setting not only the type remains unchanged, but also the function f_B. Note that a truncation setoid can be defined within the theory of setoids. In the following definition though, it is essential that $x \simeq_{\mathcal{A}} y$ is an arbitrary type.

DEFINITION 3.4.7. If (A, a_0) is a pointed type, the *suspension typoid*

$$\Sigma A = (\mathbf{2}, \simeq_{\Sigma A}, \mathtt{eq}_{\Sigma A}, *_{\Sigma A}, ^{-1_{\Sigma A}}, \cong_{\Sigma A})$$

of A is defined as follows:

$$0 \simeq_{\Sigma A} 1 \equiv \sum_{f:2\to A} f(0) =_A a_0$$

$$1 \simeq_{\Sigma A} 0 \equiv \sum_{g:2\to A} g(1) =_A a_0$$

$$0 \simeq_{\Sigma A} 0 \equiv \mathbf{1} \equiv 1 \simeq_{\Sigma A} 1$$

$$\mathtt{merid} : A \to 0 \simeq_{\Sigma A} 1$$

$$\mathtt{merid}(x) \equiv (f_x, \mathtt{refl}_{a_0})$$

$$f_x(0) \equiv a_0, \quad f_x(1) \equiv x$$

One can prove the following version of the recursion theorem for the higher inductive type ΣA of the suspension of A such that propositional equality is replaced by judgemental equality.

PROPOSITION 3.4.8. *Let $\mathcal{B}$ be a typoid, $b_0, b_1 : B$, $m : A \to b_0 \simeq_{\mathcal{B}} b_1$, and let $f : \mathbf{2} \to B$ such that $f(0) \equiv b_0$ and $f(1) \equiv b_1$. Then f is a typoid function from ΣA to $\mathcal{B}$ with an 1-associate Φ_f satisfying*

$$\Phi_f(0, 1, \mathtt{merid}(x)) \equiv m(x),$$

for every $x : A$.

Bibliography

[1] P. Aczel, M. Rathjen: *Constructive Set Theory*, manuscript, 2010.

[2] M. Artin, A. Grothendieck, J,-L. Verdier. Univers. Séminaire de GÌeométrie Algébrique du Bois Marie - 1963-64 - Théorie des topos et cohomologie étale des schémas - (SGA 4) - vol. 1, LNM 269, 185-217, Berlin; New York: Springer-Verlag.

[3] S. Awodey: *Category Theory*, Oxford University Press, 2010.

[4] G. Barthe, V. Capretta: Setoids in type theory, JFP 13 (2): 261-293, 2003.

[5] E. Bishop: *Foundations of Constructive Analysis*, McGraw-Hill, 1967.

[6] T. Coquand: A remark on singleton types, manuscript, 2014.

[7] T. Coquand, A. Spiwack: Towards Constructive Homological Algebra in Type Theory, in M. Kauers et.al (Eds.) Towards Mechanized Mathematical Assistants, Lecture Notes in Computer Science, vol 4573. Springer, Berlin, Heidelberg, 2007, 40-54.

[8] T. Coquand, N. A. Danielsson, M. H. Escardó, U. Norell, C. Xu: Negative consistent axioms can be postulated without loss of continuity, unpublished note, 2013.

[9] M. Escardó: Using Yoneda rather than J to present the identity type, Agda file, in http://www.cs.bham.ac.uk/∼mhe/yoneda/yoneda.html

[10] P. Martin-Löf: An intuitionistic theory of types: predicative part, in H. E. Rose and J. C. Shepherdson (Eds.) *Logic Colloquium'73*, pp.73-118, North-Holland, 1975.

[11] P. Martin-Löf: *Intuitionistic type theory: Notes by Giovanni Sambin on a series of lectures given in Padua, June 1980*, Napoli: Bibliopolis, 1984.

[12] E. Palmgren: Bishop's set theory, TYPES summer school Göteborg, August 2005.

[13] C. Paulin-Mohring: Inductive Definitions in the System Coq - Rules and Properties, in M. Bezem, J. F. Groote (Eds.) Proceedings of TLCA, LNM 664, Springer, 1993.

[14] E. Rijke: *Homotopy Type Theory*, Master Thesis, Utrecht University 2012.

[15] D. Scott: Constructive validity, in M. Laudet et.al. (Eds.) Symposium on Automatic demonstration, Lecture Notes in Mathematics 125, Springer, 1970, 237-235.

[16] The Univalent Foundations Program: *Homotopy Type Theory: Univalent Foundations of Mathematics*, Institute for Advanced Study, Princeton, 2013.