About the setoid model

Introduction

We present a generalization to dependent types of Gandy's interpretation of extensional in intensional type theory [2]. This can also be seen as variation of the setoid model considered in Hofmann's thesis [3]. Contrary to this model [3], the rule of pushing a substitution under an abstraction is not validated in our interpretation. The first published version of type theory [5, 6] had also this restriction and we provide thus a model of this version of type theory. On the other hand, we do interpret, contrary to [3], most computation rules as definitional equality. The only exception is the computation rule for identity type.

Our interpretation can be seen as a syntactic transformation of a type theory with extensional equality into an intensional type system (even without identity type).

Beside interpreting extensional equality, we can interpret a type of impredicative propositions. We formulate an interpretation of this new type system into an *inconsistent* type system (with a type of all types). This should provide a computational interpretation of the *axiom of unique choice*. An application may be to give computational interpretation of the definition of quotient using equivalence classes. We conjecture that the terms that we get from this interpretation are all normalizing. (This conjecture is motivated by a consistency result of univalence and resizing, which shows that the interpreted system is consistent.)

1 The setoid model

We shall model a version of type theory with extensional type theory in *intensional* type theory (this target type theory does not even need to have an identity type).

Contexts will be interpreted as *setoids*.

A setoid is a type Γ with a relation R of type $\Gamma \to \Gamma \to \mathsf{Type}$. We write $\rho_0 \rightsquigarrow_\Gamma \rho_1$ instead of $R\ \rho_0\ \rho_1$ for $\rho_0\ \rho_1 : \Gamma$. We assume that we have an operation

$$\eta_\Gamma : \prod_{\rho:\Gamma} \rho \rightsquigarrow_\Gamma \rho$$

This expresses that the given relation is reflexive. We express also that it is symmetric and transitive in a way similar to Kan extensions[1] by requiring operations of type

$$\prod_{\rho_0\ \rho_1\ \rho_2:\Gamma} \rho_0 \rightsquigarrow_\Gamma \rho_1,\ \rho_1 \rightsquigarrow_\Gamma \rho_2 \quad\longrightarrow\quad \rho_0 \rightsquigarrow_\Gamma \rho_2$$

$$\prod_{\rho_0\ \rho_1\ \rho_2:\Gamma} \rho_0 \rightsquigarrow_\Gamma \rho_2,\ \rho_1 \rightsquigarrow_\Gamma \rho_2 \quad\longrightarrow\quad \rho_0 \rightsquigarrow_\Gamma \rho_1$$

$$\prod_{\rho_0\ \rho_1\ \rho_2:\Gamma} \rho_0 \rightsquigarrow_\Gamma \rho_1,\ \rho_0 \rightsquigarrow_\Gamma \rho_2 \quad\longrightarrow\quad \rho_1 \rightsquigarrow_\Gamma \rho_2$$

A *setoid morphism* $\sigma, \sigma' : \Delta \to \Gamma$ is given by a function $\sigma : \Delta \to \Gamma$ together with a proof that it preserves the given relations

$$\sigma' : \prod_{\nu_0\ \nu_1:\Delta} \nu_0 \rightsquigarrow_\Delta \nu_1 \quad\longrightarrow\quad \sigma\nu_0 \rightsquigarrow_\Gamma \sigma\nu_1$$

[1] Our model should generalize to express groupoids, and higher-order structures.

Contrary to [3], we *don't* require the definional equality $\sigma'\ \nu\ \nu\ (\eta\nu) = \eta(\sigma\nu) : \sigma\nu \rightsquigarrow \sigma\nu$.

A dependent type $\Gamma \vdash A$ is given by a dependent type $A\rho$ over $\rho : \Gamma$ together with an operation

$$\prod_{\rho_0\ \rho_1:\Gamma}\ \prod_{\alpha:\rho_0\rightsquigarrow\rho_1} A\rho_0 \to A\rho_1 \to \mathsf{Type}$$

We write $u_0 \rightsquigarrow_\alpha u_1$ the result of this operation for $u_0 : A\rho_0,\ u_1 : A\rho_1$.

We ask furthermore to have an operation $\eta : \Pi\rho : \Gamma.\Pi u : A\rho.u \rightsquigarrow_{\eta\rho} u$ and extension operations, given $\alpha_{ij} : \rho_i \rightsquigarrow \rho_j$ for $0 \leqslant i < j \leqslant 2$

$$u_0 \rightsquigarrow_{\alpha_{01}} u_1,\ u_1 \rightsquigarrow_{\alpha_{12}} u_2 \quad\longrightarrow\quad u_0 \rightsquigarrow_{\alpha_{02}} u_2$$

$$u_0 \rightsquigarrow_{\alpha_{02}} u_2,\ u_1 \rightsquigarrow_{\alpha_{12}} u_2 \quad\longrightarrow\quad u_0 \rightsquigarrow_{\alpha_{01}} u_1$$

$$u_0 \rightsquigarrow_{\alpha_{01}} u_1,\ u_0 \rightsquigarrow_{\alpha_{02}} u_2 \quad\longrightarrow\quad u_1 \rightsquigarrow_{\alpha_{12}} u_2$$

where each of these expressions should be prefixed by a product over

$$\rho_0\ \rho_1\ \rho_2 : \Gamma,\ \alpha_{01} : \rho_0 \rightsquigarrow \rho_1,\ \alpha_{12} : \rho_1 \rightsquigarrow \rho_2,\ \alpha_{02} : \rho_0 \rightsquigarrow \rho_2,\ u_0 : A\rho_0,\ u_1 : A\rho_1,\ u_2 : A\rho_2$$

We require also a function $A\alpha^+ : A\rho_0 \to A\rho_1$ with a proof $A\alpha \uparrow u : u \rightsquigarrow_\alpha A\alpha^+ u$ and a function $A\alpha^- : A\rho_1 \to A\rho_0$ with a proof $A\alpha \downarrow u : A\alpha^- u \rightsquigarrow_\alpha u$.

These definitions can be seen as a restriction to the case of setoids of the notion of Kan simplicial set and Kan fibrations.

We define $\Gamma \vdash a : A$ to be a pair of a section $a\rho : A\rho$ for $\rho : \Gamma$ together with $a\alpha : a\rho_0 \rightsquigarrow_\alpha a\rho_1$ for $\alpha : \rho_0 \rightsquigarrow \rho_1$. We *don't* require to have the definitional equality $a\eta\rho = \eta(a\rho) : a\rho \rightsquigarrow_{\eta\rho} a\rho$.

Given $\Gamma \vdash A$ it is possible to define a new setoid $\Gamma.A$ by defining $\alpha, \omega : (\rho_0, u_0) \rightsquigarrow (\rho_1, u_1)$ iff $\alpha : \rho_0 \rightsquigarrow \rho_1$ and $\omega : u_0 \rightsquigarrow_\alpha u_1$. We can then take $\eta(\rho, u) = \eta\rho, \eta u : (\rho, u) \rightsquigarrow (\rho, u)$.

Given $\Gamma \vdash A$ and $\Gamma.A \vdash B$ it is then possible to define $\Gamma \vdash \Pi\ A\ B$ and $\Gamma \vdash \Sigma\ A\ B$. For $\rho : \Gamma$ we take $(\Pi\ A\ B)\rho$ to be the type of pairs f, f' such that $f\ u : B(\rho, u)$ for $u : A\rho$ and $f'\ u_0\ u_1\ \omega : f\ u_0 \rightsquigarrow_{\eta\rho,\omega} f\ u_1$ if $\omega : u_0 \rightsquigarrow_{\eta\rho} u_1$. We define then $(f, f') \rightsquigarrow_\alpha (g, g')$ to be

$$\prod_{u_0:A\rho_0}\ \prod_{u_1:A\rho_1} u_0 \rightsquigarrow_\alpha u_1 \quad\longrightarrow\quad f\ u_0 \rightsquigarrow_{\alpha,\omega} g\ u_1$$

We can then define $\eta(f, f') = f' : (f, f') \rightsquigarrow_{\eta\rho} (f, f')$. We take $(u, v) : (\Sigma\ A\ B)\rho$ to mean $u : A\rho$ and $v : B(\rho, u)$ while $(\omega, \delta) : (u_0, v_0) \rightsquigarrow_\alpha (u_1, v_1)$ iff $\omega : u_0 \rightsquigarrow_\alpha u_1$ and $\delta : v_0 \rightsquigarrow_{\alpha,\omega} v_1$.

We can then define $(\Pi\ A\ B) \uparrow$, $(\Pi\ A\ B) \downarrow$ and $(\Sigma\ A\ B) \uparrow$, $(\Sigma\ A\ B) \downarrow$. (See Appendix 2 for a justification of the dependent product.)

We can interpret identity type given $\Gamma \vdash A$ and $\Gamma \vdash a : A$ and $\Gamma \vdash u : A$. We take $(\mathsf{Id}_A\ a\ u)\rho$ to be the type $a\rho \rightsquigarrow_{\eta\rho} u\rho$. Given $\omega_0 : (\mathsf{Id}_A\ a\ u)\rho_0$ and $\omega_1 : (\mathsf{Id}_A\ a\ u)\rho_1$ we define $\omega_0 \rightsquigarrow_\alpha \omega_1$ to be the unit type N_1. The elimination rule over this type gets validated. However the computation rule holds only in the interpretation as a propositional equality (like in [3]).

We can also interpret a type of natural numbers by taking $\mathsf{N}\rho$ to be the type N of the target system and defining

$$0 \rightsquigarrow_{\mathsf{N}} 0 = \mathsf{N}_1 \qquad S\ n \rightsquigarrow_{\mathsf{N}} S\ m = n \rightsquigarrow_{\mathsf{N}} m \qquad 0 \rightsquigarrow_{\mathsf{N}} S\ m = \mathsf{N}_0 \qquad S\ n \rightsquigarrow_{\mathsf{N}} 0 = \mathsf{N}_0$$

where N_1 is the unit type and N_0 the empty type. Contrary to what happens in the model [3] we expect the computation rules for N to hold definitionally in this interpretation.

2 Interpretation as translation

It is then possible to check that we get a model of type theory given by the rules in Figure 1. What is *not* validated by this model are the rules of substitution under a binder. (See Appendix 1 for a simple explanation of why this rule fails in this model.) Following the discussions in [5, 6] we do not consider this to be a problem.

We can see this interpretation as a syntactic translation from an extensional type theory into an intensional type theory (without any identity type). A definitional equality $\Gamma \vdash a = u : A$ is interpreted in the target type system as a pair of definitional equalities

$$\rho : \Gamma \vdash a\rho = u\rho : A\rho \qquad \rho_0\ \rho_1 : \Gamma, \alpha : \rho_0 \rightsquigarrow \rho_1 \vdash a\alpha = u\alpha : a\rho_0 \rightsquigarrow a\rho_1$$

while a definitional equality $\Gamma \vdash A = B$ is interpreted as a pair of definitional equalities

$$\rho : \Gamma \vdash A\rho = B\rho \qquad \rho_0\ \rho_1 : \Gamma, \alpha : \rho_0 \rightsquigarrow \rho_1, u_0 : A\rho_0, u_1 : A\rho_1 \vdash u_0 \rightsquigarrow_{A\alpha} u_1 = u_0 \rightsquigarrow_{B\alpha} u_1$$

3 A type of proposition and the resizing axiom

We define a setoid Ω of propositions. The underlying type is a type of all types U while $X \rightsquigarrow_\Omega Y$ is the type $X \leftrightarrow Y$, i.e. the type $(X \to Y) \times (Y \to X)$. We define a dependent type $\Omega \vdash T$ by taking TX to be the type X while $u \rightsquigarrow_\alpha v$ is the unit type N_1 for $\alpha : X \rightsquigarrow_\Omega Y$ and $u : X$ and $v : Y$. (We can avoid to consider inconsistent type system if we are interested only in a predicative type of small propositions.)

We get in this way an interpretation of a weak form of univalence

$$\Pi\varphi_0\ \varphi_1 : \Omega.(T\varphi_0 \leftrightarrow T\varphi_1) \leftrightarrow \mathsf{Id}_\Omega\ \varphi_0\ \varphi_1$$

It is then possible to define the operations $\Gamma \vdash \forall\ A\ \varphi : \Omega$ given $\Gamma \vdash A$ and $\Gamma.A \vdash \varphi : \Omega$, with the logical equivalence $T(\forall\ A\ \varphi) \leftrightarrow \Pi\ A\ T(\varphi)$. We take

$$(\forall\ A\ \varphi)\rho = \prod_{u:A\rho} \varphi(\rho, u)$$

while, if $\alpha : \rho_0 \rightsquigarrow \rho_1$, we define

$$(\forall\ A\ \varphi)\alpha : (\prod_{u:A\rho_0} \varphi(\rho_0, u)) \leftrightarrow (\prod_{u:A\rho_1} \varphi(\rho_1, u))$$

using the maps $A\alpha^+$ and $A\alpha^-$. We can also define an operation $(\wedge) : \Omega \to \Omega \to \Omega$ with the logical equivalence $T(\varphi_0 \wedge \varphi_1) \leftrightarrow T\varphi_0 \times T\varphi_1$.

We can define $\Gamma \vdash \Lambda b : T(\forall\ A\ \varphi)$ if $\Gamma.A \vdash b : T\varphi$ by taking $(\Lambda b)\rho\ u = b(\rho, u)$ and $(\Lambda b)\alpha = 0$.

For $\Gamma \vdash A$ we have $\Gamma \vdash eq_A : A \to A \to \Omega$ such that the definitional equality $\Gamma \vdash T(eq_A\ a\ u) = \mathsf{Id}_A\ a\ u$ holds for $\Gamma \vdash a : A$ and $\Gamma \vdash u : A$.

We can also define $\Gamma \vdash \exists\ A\varphi : \Omega$ given $\Gamma \vdash A$ and $\Gamma.A \vdash \varphi : \Omega$.

For the target type system, we need a type of all types in order to interpret the existential quantification as a strong sum

$$(\exists\ A\varphi)\rho = \sum_{u:A\rho} \varphi(\rho, u)$$

What is spectacular in this interpretation is that we get an interpretation of unique choice, i.e. a proof of the following implication

$$T(\exists\ A\ \varphi) \to (\Pi a_0\ a_1 : A.\varphi[a_0] \to \varphi[a_1] \to \mathsf{Id}_A\ a_0\ a_1) \to \Sigma\ A\ T(\varphi)$$

For this, it is crucial that we interpret existential quantification as a sigma type,

In this type system, it should be possible to define the quotient type using equivalence classes, and for instance, to define $\mathbb{Z}$ as a quotient of $\mathbb{N} \times \mathbb{N}$. Voevodsky had such a development using resizing rules and equivalence and it would be interesting to see if we get a natural computational content of this development in this way.

$$\frac{\Gamma \vdash}{1 : \Gamma \to \Gamma} \qquad \frac{\sigma : \Delta \to \Gamma \quad \delta : \Theta \to \Delta}{\sigma\delta : \Theta \to \Gamma}$$

$$\frac{\Gamma \vdash A \qquad \sigma : \Delta \to \Gamma}{\Delta \vdash A\sigma} \qquad \frac{\Gamma \vdash t : A \qquad \sigma : \Delta \to \Gamma}{\Delta \vdash t\sigma : A\sigma}$$

$$\frac{}{\vdash} \qquad \frac{\Gamma \vdash \qquad \Gamma \vdash A}{\Gamma.A \vdash} \qquad \frac{\Gamma \vdash A}{\mathsf{p} : \Gamma.A \to \Gamma} \qquad \frac{\Gamma \vdash A}{\Gamma.A \vdash \mathsf{q} : A\mathsf{p}}$$

$$\frac{\sigma : \Delta \to \Gamma \quad \Gamma \vdash A \quad \Delta \vdash u : A\sigma}{(\sigma, u) : \Delta \to \Gamma.A}$$

$$\frac{\Gamma.A \vdash B}{\Gamma \vdash \Pi\ A\ B} \qquad \frac{\Gamma.A \vdash B \quad \sigma : \Delta \to \Gamma \quad \Delta.A\sigma \vdash b : B(\sigma\mathsf{p}, \mathsf{q})}{\Delta \vdash \lambda b : (\Pi\ A\ B)\sigma}$$

$$\frac{\Gamma.A \vdash B}{\Gamma \vdash \Sigma\ A\ B} \qquad \frac{\Gamma.A \vdash B \quad \sigma : \Delta \to \Gamma \quad \Delta \vdash u : A\sigma \quad \Delta \vdash v : B(\sigma, u)}{\Delta \vdash (u, v) : (\Sigma\ A\ B)\sigma}$$

$$\frac{\sigma : \Delta \to \Gamma \qquad \Delta \vdash w : (\Pi\ A\ B)\sigma \qquad \Delta \vdash u : A\sigma}{\Delta \vdash \mathsf{app}(w, u) : B(\sigma, u)}$$

$$\frac{\Delta \vdash w : (\Sigma\ A\ B)\sigma}{\Delta \vdash \mathsf{p}w : A\sigma} \qquad \frac{\Delta \vdash w : (\Sigma\ A\ B)\sigma}{\Delta \vdash \mathsf{q}w : B(\sigma, \mathsf{p}w)}$$

$$\sigma 1 = \sigma \qquad 1\sigma = \sigma \qquad (\sigma\delta)\nu = \sigma(\delta\nu)$$

$$(\sigma, u)\delta = (\sigma\delta, u\delta) \qquad \mathsf{p}(\sigma, u) = \sigma \qquad \mathsf{q}(\sigma, u) = u$$

$$\mathsf{app}(w, u)\delta = \mathsf{app}(w\delta, u\delta) \qquad \mathsf{app}((\lambda b)\sigma, u) = b(\sigma, u)$$

Figure 1: Rules of WMLTT

Appendix 0: WMLTT

We call WMLTT the version of Type Theory with rules are presented in Figure 1. This is equivalent to the version of type theory presented in the references [5, 6].

For the typing rules, we remove the conversion rule $(\Pi\ A\ B)\sigma = \Pi\ (A\sigma)\ (B(\sigma\mathsf{p}, \mathsf{q}))$ and have instead the following rules

$$\frac{\Gamma \vdash A \quad \Gamma.A \vdash B \qquad \sigma : \Delta \to \Gamma \qquad \Delta \vdash w : (\Pi\ A\ B)\sigma \qquad \Delta \vdash u : A\sigma}{\Delta \vdash \mathsf{app}(w, u) : B(\sigma, u)}$$

and the conversion rule is

$$\frac{\Gamma \vdash A \quad \Gamma.A \vdash B \qquad \sigma : \Delta \to \Gamma \qquad \Gamma.A \vdash b : B \qquad \Delta \vdash u : A\sigma}{\Delta \vdash \mathsf{app}((\lambda b)\sigma, u) = b(\sigma, u) : B(\sigma, u)}$$

Appendix 1: The setoid model for simple type theory

A type A is a pair (X, R) where X is a set and $R\ x\ u$ a relation over X which is reflexive, symmetric and transitive. In particular for any x in X we have an element $\eta(x) : R\ x\ x$. We define

$$(X, R) \to (Y, S) = (Z, T)$$

where Z is the set of pairs f, f' with $f : X \to Y$ and $f' : \Pi\ x\ u : X.R\ x\ u \to S\ (f\ x)\ (f\ u)$, and

$$T\ (f, f')\ (g, g') = \Pi\ x\ u : X.R\ x\ u \to S\ (f\ x)\ (g\ u)$$

It is rather direct to show that T is reflexive, symmetric and transitive with $\eta(f, f') = f'$.

The following notation will be convenient. If $t : (X, R) \to (Y, S)$ with $t = (f, f')$ and $\rho : X$ we write $t\rho$ for $f\ \rho : Y$ and if $\alpha : R\ \rho_0\ \rho_1$ we write $t\alpha : S\ (t\rho_0)\ (t\rho_1)$ for $f'\ \rho_0\ \rho_1\ \alpha$.

We define $(X, R) \times (Y, S) = (X \times Y, T)$ with $T\ (x, y)\ (u, v) = R\ x\ u \times S\ y\ v$. It is direct how to define $\mathsf{p} : A \times B \to A$ and $\mathsf{q} : A \times B \to B$.

If $t : (X, R) \to ((Y, S) \to (Z, T))$ and $u : (X, R) \to (Y, S)$ we define $\mathsf{app}(t, u) : (X, R) \to (Z, T)$ by taking

$$\mathsf{app}(t, u)\rho = t\rho\ (u\rho) \qquad \mathsf{app}(t, u)\alpha = t\alpha\ (u\alpha)$$

We define $\lambda(c) : A \to (B \to C)$ for $c : A \times B \to C$. This is defined by

$$\lambda(c)\rho\nu = c(\rho, \nu) \qquad \lambda(c)\rho\beta = c(\eta(\rho), \beta)$$

and

$$\lambda(c)\alpha\beta = c(\alpha, \beta)$$

Given $\sigma : D \to A$ we can now compare

$$\lambda(c)\sigma\ d = (\lambda y.c(\sigma\ d, y), \lambda q.c(\eta(\sigma\ d), q))$$

and

$$\lambda(c(\sigma\mathsf{p}, \mathsf{q}))\ d = (\lambda y.c(\sigma\ d, y), \lambda q.c(\sigma(\eta(d)), q))$$

They are equal only if

$$\sigma(\eta(d)) = \eta(\sigma\ d)$$

and this is not valid in general.

On the other hand

$$\mathsf{app}(\lambda(c)\sigma, u) = c(\sigma, u)$$

is valid. This is because we have

$$\mathsf{app}(\lambda(c)\sigma, u)\ d = c(\sigma\ d, u\ d)$$

and

$$\mathsf{app}(\lambda(c)\sigma, u)\ \alpha = c(\sigma\ \alpha, u\ \alpha)$$

Appendix 2: Justification of the dependent product

If $\Gamma \vdash A$ and $\Gamma.A \vdash B$ we define $\Gamma \vdash \Pi\ A\ B$. If $\rho : \Gamma$ then $(\Pi\ A\ B)\rho$ is the set of pairs f, f' with $f\ u : B(\rho, u)$ if $u : A\rho$ and $f'\ \omega : f\ u_0 \to f\ u_1$ over $\eta(\rho), \omega$ whenever $\omega : u_0 \to_{\eta\rho} u_1$. If $\alpha : \rho_0 \to \rho_1$ and $f_0, f_0' : (\Pi\ A\ B)\rho_0,\ f_1, f_1' : (\Pi\ A\ B)\rho_1$ we define $\lambda : f_0 \to_\alpha f_1$ to be a function sending $\omega : u_0 \to_\alpha u_1$ to $\lambda\ \omega : f_0\ u_0 \to_{\alpha,\omega} f_1\ u_1$. We define $\eta(f, f') = f'$.

If $f, f' : (\Pi\ A\ B)\rho_0$ and $\alpha : \rho_0 \to \rho_1$, we define $(\Pi\ A\ B)\alpha^+(f, f') = g, g' : (\Pi\ A\ B)\rho_1$ in the following way. First $g\ v = B(\alpha, A\alpha \downarrow v)^+(f\ (A\alpha^- v))$ is in $B(\rho_1, v)$ if $v : A\rho_1$. If we have $v \to v'$ we have $g\ v \to g\ v'$ which defines g'. To define a proof of $f, f' \to_\alpha g, g'$ we take $u \to v$. We have then a triangle $u \to A\alpha^- v,\ A\alpha^- v \to v, u \to v$ and we deduce $f\ u \to f\ (A\alpha^- v)$ and then $f\ u \to g\ v$.

Given $\theta = (\alpha_{01}, \alpha_{12}, \alpha_{02})$ and $\lambda_{01} : f_0 \to_{\alpha_{01}} f_1,\ \lambda_{02} : f_0 \to_{\alpha_{02}} f_2$ we define $\mathsf{comp}_0(\theta, \lambda_{01}, \lambda_{02}) : f_1 \to_{\alpha_{12}} f_2$. For this, we take $u_1 \to_{A\alpha_{12}} u_2$. We then have a triangle $\nu = (\omega_{01}, \omega_{12}, \omega_{02})$ with $\omega_{01} = A\alpha_{01} \downarrow u_1 : A\alpha_{01}^- u_1 \to u_1,\ \omega_{12} : u_1 \to u_2,\ \omega_{02} : A\alpha_{01}^- u_1 \to u_2$. Hence we get $\mathsf{comp}_0((\theta, \nu), \lambda_{01}\ \omega_{01}, \lambda_{02}\ \omega_{02}) : f_1\ u_1 \to f_2\ u_2$.

Appendix 3: Justification of the abstraction rule

The abstraction rule is

$$\frac{\Gamma.A \vdash B \quad \sigma : \Delta \to \Gamma \quad \Delta.A\sigma \vdash b : B(\sigma\mathsf{p}, \mathsf{q})}{\Delta \vdash \lambda b : (\Pi\ A\ B)\sigma}$$

Given $\nu : \Delta$ we have to define $(\lambda b)\nu$ in $(\Pi\ A\ B)\sigma\nu$. An element of this type is a pair. For the first element we take the function $f\ u = b(\nu, u) : B(\sigma\nu, u)$ for $u : A\sigma\nu$. The second element f' takes as

argument $u_0,\ u_1 : A\sigma\nu$ and a proof $\omega : u_0 \to_{\eta\sigma\nu} u_1$ and should produce an element $f'\ u_0\ u_1\ \omega : b(\nu, u_0) \rightsquigarrow_{\eta\sigma\nu,\omega} b(\nu, u_1)$. Using $\Gamma \vdash A$ and the triangle $\eta\sigma\nu,\ \eta\sigma\nu,\ \sigma\eta\nu$ we build $\omega' : u_0 \to_{\sigma\eta\nu} u_1$. Using $\Delta.A\sigma \vdash b : B(\sigma\mathsf{p}, \mathsf{q})$ we get $b(\eta\nu, \omega') : b(\nu, u_0) \rightsquigarrow_{\sigma\eta\nu,\omega'} b(\nu, u_1)$ and then using $\Gamma.A \vdash B$ and the triangle $(\sigma\eta\nu, \omega'),\ (\eta\sigma\nu, \omega),\ (\eta\sigma\nu, \eta u_1)$ we get an element in $b(\nu, u_0) \rightsquigarrow_{\eta\sigma\nu,\omega} b(\nu, u_1)$ as required.

We need also to define a function sending $\alpha : \nu_0 \rightsquigarrow \nu_1$ and $\omega : u_0 \rightsquigarrow_{\sigma\alpha} u_1$ to $b(\nu_0, u_0) \rightsquigarrow_{\sigma\alpha,\omega} (\nu_1, u_1)$. We take $b(\alpha, \omega) : b(\nu_0, u_0) \rightsquigarrow_{\sigma\alpha,\omega} (\nu_1, u_1)$.

References

[1] J. Cartmell. Generalised algebraic theories and contextual categories. Ann. Pure Appl. Logic 32 (1986), no. 3, 209–243.

[2] R. Gandy. On The Axiom of Extensionality -Part I. The Journal of Symbolic Logic, Vol. 21, 1956.

[3] M. Hofmann. Extensional concepts in intensional type theory. Ph.D. thesis, Edinburgh, 1994.

[4] J. Lambek and P.J. Scott. *Introduction to higher order categorical logic.* Cambridge studies in advanced mathematics 7, 1986.

[5] P. Martin-Löf. An intuitionistic theory of types: predicative part. Logic Colloquium, 1973.

[6] P. Martin-Löf. About models for intuitionistic type theories and the notion of definitional equality Proceedings of the Third Scandinavian Symposium, North-Holland, 1975.

[7] V. Voevodsky. Univalent foundations project. NSF grant application, 2010.