Canonicity and normalization for Dependent Type Theory

Introduction

We show canonicity and normalization for dependent type theory with a cumulative sequence of universes $U_1 : U_2 \ldots$ with η-conversion. We first give the argument in a constructive set theory (CZF extended with universes) and then as a program transformation in an extension of type theory. The argument is then "optimal" in the sense that the metatheory used in the argument is as close as possible to the object theory.

Note that subject reduction is not clear since we consider a system with conversion as judgements) and we don't introduce neither a reduction relation in the syntax nor a logical relation in the semantics.

1 Type system

We use conversion as judgements [1, 2]. It is not clear a priori that subject reduction holds.

$$\frac{\Gamma \vdash A : U_n}{\Gamma, x : A \vdash} \qquad \frac{}{() \vdash} \qquad \frac{\Gamma \vdash}{\Gamma \vdash x : A}\ (x{:}A\ in\ \Gamma)$$

$$\frac{\Gamma \vdash A : U_n \qquad \Gamma, x : A \vdash B : U_n}{\Gamma \vdash \Pi(x : A)B : U_n} \qquad \frac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \lambda(x : A)t : \Pi(x : A)B} \qquad \frac{\Gamma \vdash t : \Pi(x : A)B \qquad \Gamma \vdash u : A}{\Gamma \vdash t\ u : B(u)}$$

$$\frac{\Gamma \vdash A : U_n}{\Gamma \vdash A : U_m}\ (n \leqslant m) \qquad \frac{}{\Gamma \vdash U_n : U_m}\ (n < m) \qquad \frac{}{\Gamma \vdash N_2 : U_n}$$

The conversion rules are

$$\frac{\Gamma \vdash t : A \qquad \Gamma \vdash A\ \mathsf{conv}\ B : U_n}{\Gamma \vdash t : B} \qquad \frac{\Gamma \vdash t\ \mathsf{conv}\ u : A \qquad \Gamma \vdash A\ \mathsf{conv}\ B : U_n}{\Gamma \vdash t\ \mathsf{conv}\ u : B}$$

$$\frac{\Gamma \vdash t : A}{\Gamma \vdash t\ \mathsf{conv}\ t : A} \qquad \frac{\Gamma \vdash t\ \mathsf{conv}\ v : A \qquad \Gamma \vdash u\ \mathsf{conv}\ v : A}{\Gamma \vdash t\ \mathsf{conv}\ u : A}$$

$$\frac{\Gamma \vdash A\ \mathsf{conv}\ B : U_n}{\Gamma \vdash A\ \mathsf{conv}\ B : U_m}\ (n \leqslant m) \qquad \frac{\Gamma \vdash A_0\ \mathsf{conv}\ A_1 : U_n \qquad \Gamma, x : A_0 \vdash B_0\ \mathsf{conv}\ B_1 : U_n}{\Gamma \vdash \Pi(x : A_0)B_0\ \mathsf{conv}\ \Pi(x : A_1)B_1 : U_n}$$

$$\frac{\Gamma \vdash t\ \mathsf{conv}\ t' : \Pi(x : A)B \qquad \Gamma \vdash u : A}{\Gamma \vdash t\ u\ \mathsf{conv}\ t'\ u : B(u)} \qquad \frac{\Gamma \vdash t : \Pi(x : A)B \qquad \Gamma \vdash u\ \mathsf{conv}\ u' : A}{\Gamma \vdash t\ u\ \mathsf{conv}\ t\ u' : B(u)}$$

$$\frac{\Gamma, x : A \vdash t : B \qquad \Gamma \vdash u : A}{\Gamma \vdash (\lambda(x : A)t)\ u\ \mathsf{conv}\ t(u) : B(u)}$$

We consider type theory with η-rules

$$\frac{\Gamma \vdash t : \Pi(x : A)B \quad \Gamma \vdash u : \Pi(x : A)B \quad \Gamma, x : A \vdash t\ x\ \mathsf{conv}\ u\ x : B}{\Gamma \vdash t\ \mathsf{conv}\ u : \Pi(x : A)B}$$

Finally we add $N_2 : U_1$ with the rules

$$\frac{}{\Gamma \vdash 0 : N_2} \qquad \frac{}{\Gamma \vdash 1 : N_2} \qquad \frac{\Gamma, x : N_2 \vdash C : U_n \qquad \Gamma \vdash a_0 : C(0) \qquad \Gamma \vdash a_1 : C(1)}{\Gamma \vdash \mathsf{brec}\ (\lambda x.C)\ a_0\ a_1 : \Pi(x : N_2)C}$$

with computation rules $\mathsf{brec}\ (\lambda x.C)\ a_0\ a_1\ 0\ \mathsf{conv}\ a_0 : C(0)$ and $\mathsf{brec}\ (\lambda x.C)\ a_0\ a_1\ 1\ \mathsf{conv}\ a_1 : C(1)$.

We let Type_n be the set of expressions that are closed terms of type U_n *modulo* conversion. Similarly if $A \in \mathsf{Type}_n$ then $\mathsf{Elem}(A)$ the set of expressions that are closed elements of type A *modulo* conversion, so that $\mathsf{Type}_n = \mathsf{Elem}(U_n)$. If $A \in \mathsf{Type}_n$ we let $\mathsf{Term}(A)$ be the set of expressions of type A. If e is in $\mathsf{Term}(A)$ we write $[e]$ in $\mathsf{Elem}(A)$ the equivalence class of e for conversion. If $A \in \mathsf{Type}_n$ we let $\mathsf{Norm}(A)$ (resp. $\mathsf{Neut}(A)$) be the subset $\mathsf{Term}(A)$ of all expressions of type A that are in normal form (resp. neutral). Note that we have both $\mathsf{Term}(A)$ and $\mathsf{Elem}(A)$ in $\mathcal{U}_0$.

If $A \in \mathsf{Type}$ and $a \in \mathsf{Elem}(A)$ then we let $\mathsf{Norm}(A)|a$ (resp. $\mathsf{Neut}(A)|a$) be the set of expressions in $\mathsf{Norm}(A)$ (resp. $\mathsf{Neut}(A)$) convertible to a.

2 First Interpretation

We let and $\mathcal{U}_1$ be the first Grothendieck's universe, $\mathcal{U}_2$ be the next universe, and so on.

If $A \in \mathsf{Type}_n$ we let $\mathsf{Comp}_n(A)$ be the set of family of sets in $\mathcal{U}_n$ over $\mathsf{Elem}(A)$.

We define C_{N_2} in $\mathsf{Comp}(N_2)$ by taking $C_{N_2}(t)$ to be the set $\{0 \mid t \;\mathsf{conv}\; 0 : N_2\} \cup \{1 \mid t \;\mathsf{conv}\; 1 : N_2\}$.

If $A : U_n$ and $B(x) : U_n$ $(x : A)$ and C_A is in $\mathsf{Comp}_n(A)$ and we have a family $C_B(a, \nu)$ in $\mathsf{Comp}(B(a))$ for $a \in \mathsf{Elem}(A)$ and $\nu \in C_A(a)$ we define $C_T = \Pi\ C_A\ C_B$ in $\mathsf{Comp}_n(T)$ where $T = \Pi(x : A)B$ by

$$C_T(w) = \Pi(a \in \mathsf{Elem}(A))\Pi(\nu \in C_A(a))C_B(a, \nu)(w\ a)$$

To a context Γ we associate $\mathsf{Elem}(\Gamma)$ which is the set of vectors $\rho_0 = (a_1, \dots, a_n)$ with a_i in $\mathsf{Elem}(A_i(a_1, \dots, a_{i-1}))$ and a family of sets $C_\Gamma(\rho)$ defined by taking $C_{\Gamma,x:A}(\rho_0, a)$ to be the sets of vectors (ρ_1, ν) with ρ_1 in $C_\Gamma(\rho)$ and ν in $[\![A]\!]_\rho(a)$. We write $\rho = \rho_0, \rho_1$.

We define then $[\![A]\!]_\rho$ if $\Gamma \vdash A$ and $[\![a]\!]_\rho$ if $\Gamma \vdash a : A$ by the following rules

- $[\![x_i]\!]_\rho = \rho_1(x)$
- $[\![c\ a]\!]_\rho = [\![c]\!]_\rho\ a\rho_0\ [\![a]\!]_\rho$
- $[\![\lambda(x : A)t]\!]_\rho\ a\ \nu = [\![t]\!]_{(\rho_0,a),(\rho_1,\nu)}$
- $[\![\Pi(x : A)B]\!]_\rho = \Pi[\![A]\!]_\rho((a, \nu) \mapsto [\![B]\!]_{(\rho_0,a),(\rho_1,\nu)})$
- $[\![U_n]\!]_\rho = \mathsf{Comp}_n$
- $[\![N_2]\!]_\rho = C_{N_2}$

We define $[\![\mathsf{brec}\ (\lambda x.C)\ a_0\ a_1]\!]_\rho$ as a function f which takes as argument $u : N_2$ and ν in $C_{N_2}(u)$. If $\nu = 0$ we take $f\ u\ \nu = [\![a_0]\!]_\rho$ and if $\nu = 1$ we take $f\ u\ \nu = [\![a_1]\!]_\rho$.

Theorem 2.1 *If $\vdash A : U_n$ then $[\![A]\!]$ is a computability structure on A. If $\vdash a : A$ we have $[\![a]\!] \in [\![A]\!](a)$, and if $\vdash a = b : A$ we have $[\![a]\!] = [\![b]\!]$.*

Proof. We generalize the statements to open terms. If $\Gamma \vdash A$ and $C_\Gamma(\rho)$ then $[\![A]\!]_\rho$ is of the form C where $C(t)$ is a set for t in $\mathsf{Elem}(: A\rho_0)$. If $\Gamma \vdash A\ \mathsf{conv}\ B$ we have $[\![A]\!]_\rho = [\![B]\!]_\rho$. If $\Gamma \vdash a : A$ we have $[\![a]\!]_\rho$ in $C_A(\rho)\ a\rho_0$. Finally if $\Gamma \vdash a_0 = a_1 : A$ we have $[\![a_0]\!]_\rho = [\![a_1]\!]_\rho$.

We then proceed by induction on derivations. We explain the case of the rule

$$\frac{\Gamma \vdash a : A \qquad \Gamma, x : A \vdash b : B}{\Gamma \vdash (\lambda(x : A)b)\ a\ \mathsf{conv}\ b(a) : B(a)}$$

To simplify the presentation we assume Γ empty. We write $t = (\lambda(x : A)b)\ a$. By induction hypothesis we have $[\![a]\!] \in C_A(a)$ and $\xi \in C_B(u, \nu)(b(u))$ if ν is in $[\![A]\!](u)$ where $C_A = [\![A]\!]$ and $\xi = [\![b]\!]_{(u,\nu)}$. We can take $u = a$ and $\nu = [\![a]\!]$, and then, by the convertibility we have ξ in $C_B(t)$. □

Corollary 2.2 *If $\vdash t : N_2$ we have $t\ \mathsf{conv}\ 0 : N_2$ or $t\ \mathsf{conv}\ 1 : N_2$. Furthermore, we cannot have $0\ \mathsf{conv}\ 1 : N_2$.*

3 Canonicity argument as a program transformation

If one looks at the argument used for the proof of canonicity one can see that the proof itself can directly be expressed in an extension of type theory. Note that we use *definitional conversion* in the meta theory in order to interpret conversion in the object theory, as advocated in [5] (with the crucial difference that we allow η-conversion at both levels).

We write $\Gamma \vdash_0 a : A$ for the judgement of the *object* type theory. The extension is as follows. It is like the object theory, and we have a cumulative hierarchy of universes $\mathcal{U}_0, \mathcal{U}_1, \dots$ but we also have *new* base types Type_n and $\mathsf{Elem}(A)$ for A in Type_n. The introduction rules are the following

$$\frac{\vdash_0 A : U_n}{\Gamma \vdash \mathsf{Elem}(A) \in \mathcal{U}_0} \qquad \frac{\vdash_0 a : A}{\Gamma \vdash [a] \in \mathsf{Elem}(A)}$$

A context in this extension has the form $\xi_1 \in T_1, \dots, \xi_n \in T_n$. If w is in $\mathsf{Elem}(\Pi(x : A)B)$ and u is in $\mathsf{Elem}(A)$ we have $w = [c]$ and $u = [a]$ for some expressions $\vdash_0 c : \Pi(x : A)B$ and $\vdash_0 a : A$ and we can form $w\ u = [c\ a]$ in $\mathsf{Elem}(B(a))$.

To any term $\vdash_0 A : U_n$ of the base theory we associate a term $\vdash [\![A]\!] \in \mathsf{Elem}(A) \to \mathcal{U}_n$ of the meta theory. Similarly, to any term $\vdash_0 a : A$ we associate $\vdash [\![a]\!] \in [\![A]\!]\ [a]$. Both operations can be seen as *program transformations*, simply reading the previous section in a purely syntactical way.

Here is an example. We have $\vdash t : T$ where $t = \lambda(A : U_0)(a : A)a$ and $T = \Pi(A : U_0)A \to A$. We compute

$$[\![T]\!]\ w = \Pi(A \in \mathsf{Elem}(U_0))(C_A \in \mathsf{Elem}(A) \to \mathcal{U}_0)(a \in \mathsf{Elem}(A))C_A(a) \to C_A(w\ a)$$

and

$$[\![t]\!] = \lambda(A \in \mathsf{Elem}(U_0))(C_A \in \mathsf{Elem}(A) \to \mathcal{U}_0)(a \in \mathsf{Elem}(A))(\nu \in C_A(a))\nu$$

so that $[\![t]\!]$ is an element of $[\![T]\!]\ [t]$. We use the fact that $[\![t]\!]\ a = a$ in $\mathsf{Elem}(A)$ if a is in $\mathsf{Elem}(A)$.

4 Normalization

We consider now terms in a context, and we relativize all the previous definitions to the presheaf topos over the category of contexts and weakening.

We also refine the definition of $\mathsf{Comp}_n(A)$ It is now the set of elements C, α, β where $C(t)$ is a *set* in $\mathcal{U}_n$ for each t in $\mathsf{Elem}(A)$ and β in $\Pi(k \in \mathsf{Neut}(A))C([k])$ and α in $\Pi(a \in \mathsf{Elem}(A))\ C(a) \to \mathsf{Norm}(A)|a$.

We redefine $C_{N_2}(t)$ to be the set of *syntactical expressions* u in $\mathsf{Norm}(N_2)|t$ such that u is 0 or 1 or is neutral and $\alpha_{N_2}(t, \nu) = \nu$ and $\beta_{N_2}(k) = k$.

We define $C_{U_n}(A)$ to be $\mathsf{Norm}(U_n)|A \times \mathsf{Comp}_n(A)$.

Finally $\alpha_{U_n}(A, (A', C_A, \alpha_A, \beta_A)) = A'$ and for K neutral $\beta_{U_n}(K) = (K, C, \alpha, \beta)$ where $C(t)$ is $\mathsf{Neut}([K])|t$ and $\alpha(t, k) = k$ and $\beta(k) = k$.

Assume that $A : U_n$ and $B(x) : U_n\ (x : A)$ is a family of type over A. If $E_A = (A', C_A, \alpha_A, \beta_A)$ is in $C_U(A)$ and $E_B(a, \nu) = (B'(a, \nu), C_B(a, \nu), \alpha_B(a, \nu), \beta_B(a, \nu))$ is in $C_U(B(a))$ for a in $\mathsf{Elem}(A)$ and ν in $C_A(u)$, we define $\Pi\ E_A\ E_B = (\Pi(x : A')B'(x, \beta_A(x)), C, \alpha, \beta)$ where

- $C(w) = \Pi(a \in \mathsf{Elem}(A))\Pi(\nu \in C_A(u))C_B(a, \nu)(w\ u)$
- $\beta(k)\ a\ \nu = \beta_B(a, \nu)(k\ \alpha_A(a, \nu))$
- $\alpha(w, \xi) = \lambda(x : A')\alpha_B(x, \beta_A(x))(w\ x, \xi\ x\ \beta_A(x))$

We define $[\![U_n]\!] = U_n, C_{U_n}, \alpha_{U_n}, \beta_{U_n}$ and $[\![N_2]\!] = N_2, C_{N_2}, \alpha_{N_2}, \beta_{N_2}$.

Theorem 4.1 *If $\vdash A : U_n$ then $[\![A]\!]$ is in $C_{U_n}(A)$. If $\vdash a : A$ we have $[\![a]\!]$ in $C(a)$, where $(A', C, \alpha, \beta) = [\![A]\!]$, and if a conv $b : A$ we have $[\![a]\!] = [\![b]\!]$.*

It follows that conversion is decidable: if a_0 and a_1 are of type A we can compute $[\![A]\!] = (A', C, \alpha, \beta)$ and a_0 conv $a_1 : A$ if, and only if $\alpha\ [a_0]\ [\![a_0]\!] = \alpha\ [a_1]\ [\![a_1]\!]$.

We also can prove that Π is one-to-one for conversions, following P. Hancock's argument presented in [4].

5 Normalization as program transformation

The idea is now to express type theoretically the presehaf models over the category of contexts and weakening. We introduce the types $\mathsf{Term}(A)$ of terms of type A, so that $\Delta \vdash \mathsf{Term}(A) \in \mathcal{U}_0$ if $\Delta \vdash A : \mathsf{Term}(U_n)$. We also introduce the subtypes $\mathsf{Norm}(A)$ (resp. $\mathsf{Neut}(A)$) of normal (resp. neutral) syntactical expressions representing elements of type A, as well as the subtypes $\mathsf{Var}(A)$ of variables of type A. We also have the type $\mathsf{Elem}(A)$ of elements of type A, that are expressions of type A modulo conversion, so that $\Delta \vdash \mathsf{Elem}(A) \in \mathcal{U}_0$ if $\Delta \vdash_0 A \in \mathsf{Term}(U_n)$ and we have $\Delta \vdash [a] \in \mathsf{Elem}(A)$ whenever $\Delta \vdash a \in \mathsf{Term}(A)$

We have for instance the rule

$$\frac{\Delta \vdash A \in \mathsf{Norm}(U_n) \qquad \Delta, x : \mathsf{Var}(A) \vdash B \in \mathsf{Norm}(U_n)}{\Delta \vdash \Pi(x : A)B \in \mathsf{Norm}(U_n)} \qquad \frac{\Delta \vdash A \in \mathsf{Norm}(U_n) \qquad \Delta, x : \mathsf{Var}(A) \vdash b \in \mathsf{Norm}(B)}{\Delta \vdash \lambda(x : A)b \in \mathsf{Norm}(\Pi(x : A)B)}$$

We can define $B \cdot a \in \mathsf{Elem}(U_n)$ if B is in $\mathsf{Var}(A) \to \mathsf{Elem}(U_n)$ and a in $\mathsf{Elem}(A)$ as well as $c\dot{a}$ in $\mathsf{Elem}(B \cdot a)$ if c is in $\mathsf{Elem}(\Pi(x : A)B)$ and a in $\mathsf{Elem}(A)$.

We then define

$$\mathsf{Comp}_n(A) = \Sigma(C \in \mathsf{Elem}(A) \to \mathcal{U}_n)(\Pi(a \in \mathsf{Elem}(A))(C(a) \to \mathsf{Norm}(A)|a)) \times \Pi(k \in \mathsf{Neut}(A))C([k])$$

A *computability structure* for $A \in \mathsf{Term}(U_m)$ is an element of $C_{U_m}(A) = \mathsf{Norm}(U_m)|A \times \mathsf{Comp}_m(A)$.

We define $\alpha_{U_n}(A, (A', C_A, \alpha_A, \beta_A)) = A'$ and for $K \in \mathsf{Neut}(U_m)$ we take $\beta(K) = (K, C, \alpha, \beta)$ where $C(t) = \mathsf{Neut}(K)|t$ and $\alpha(t, k) = k$ and $\beta(k) = k$.

Assume that $B(x)$ $(x : A)$ is a family of type over A. If $E_A = (A', C_A, \alpha_A, \beta_A)$ is a computability structure on A and $E_B(a, \nu) = (B'(a, \nu), C_B(a, \nu), \alpha_B(a, \nu), \beta_B(a, \nu))$ is a computability structure on $B \cdot u$ for $u \in \mathsf{Elem}(A)$ and $\nu \in C_A(u)$, we define $\Pi\ E_A\ E_B = (\Pi(x : A')B'([x], \beta_A(x)), C, \alpha, \beta)$ where

- $C(w) = \Pi(a \in \mathsf{Elem}(A))\Pi(\nu \in C_A(u))C_B(a, \nu)(w\ a)$
- $\beta(k) = \lambda(a \in \mathsf{Elem}(A))\lambda(\nu \in C_A(u))\beta_B(a, \nu)(k\ \alpha_A(a, \nu))$
- $\alpha(w, \xi) = \lambda(x : A')\alpha_B([x], \beta_A(x))(w\ [x], \xi\ [x]\ \beta_A(x))$

We can define as well $\Sigma\ E_A\ E_B = (\Sigma(x : A')B'(x, \beta_A(x)), C, \alpha, \beta)$ where

- $C(a, b) = \Sigma(\nu \in C_A(u))C_B(a, \nu)(w\ a)$
- $\beta(k) = \beta_A(k.1), \beta_B(k.1, \beta_A(k.1))(k.2)$
- $\alpha(a, b)(\nu, \delta) = \alpha_A(a, \nu), \alpha_B(a, \nu)(b, \delta)$

We then have $\Pi\ E_A\ E_B \in C_{U_n}(\Pi(x : A)B)$ if $E_A \in C_{U_n}(A)$ and $E_B \in \Pi(a : A)E_A.2\ a \to C_{U_n}(B\ a)$

We can then formulate the previous results as a program transformation in the style of [3].

If $x_1 : A_1, \ldots, x_n : A_n \vdash_0 a : A$ and we fix another context Γ of the base theory with $\Gamma \vdash a_i : A(a_1, \ldots, a_n)$ we write $\rho = \rho_0, \rho_1$ with $\rho_0 = a_1, \ldots, a_n$ and $\rho_1 = \nu_1, \ldots, \nu_n$ with $\vdash_\Gamma \nu_i \in [\![A]\!]_\rho.2\ a_i$.

The program transformation is given by

- $[\![x_i]\!]_\rho = \nu_i$
- $[\![c\ a]\!]_\rho = [\![c]\!]_\rho\ a\rho_0\ [\![a]\!]_\rho$
- $[\![\lambda(x : A)t]\!]_\rho = \lambda(a \in \mathsf{Elem}(A\rho_0))\lambda(\nu \in [\![A]\!]_\rho.2\ a)[\![t]\!]_{(\rho_0, a), (\rho_1, \nu)}$
- $[\![\Pi(x : A)B]\!]_\rho = \Pi[\![A]\!]_\rho((a, \nu) \mapsto [\![B]\!]_{(\rho_0, a), (\rho_1, \nu)})$
- $[\![\Sigma(x : A)B]\!]_\rho = \Sigma[\![A]\!]_\rho((a, \nu) \mapsto [\![B]\!]_{(\rho_0, a), (\rho_1, \nu)})$
- $[\![U_n]\!]_\rho = U_n, C_{U_n}, \alpha_{U_n}, \beta_{U_n}$
- $[\![N_2]\!]_\rho = N_2, C_{N_2}, \alpha_{N_2}, \beta_{N_2}$

Theorem 5.1 *If $x_1 : A_1, \ldots, x_n : A_n \vdash_0 a : A$ then $\vdash_\Gamma [\![a]\!]_\rho \in [\![A]\!]_\rho\ a\rho_0$ and if $x_1 : A_1, \ldots, x_n : A_n \vdash_0 a$ conv $b : A$ then $\vdash_\Gamma [\![a]\!]_\rho = [\![b]\!]_\rho \in [\![A]\!]_\rho\ a\rho_0$.*

References

[1] A. Abel and G. Scherer. On Irrelevance and Algorithmic Equality in Predicative Type Theory. In Logical Methods in Computer Science, 8(1):1-36, 2012.

[2] R. Adams. Pure type systems with judgemental equality Journal of Functional Programming. 16, 2, p. 219-246, 2006.

[3] J.-Ph. Bernardy, P. Jansson, R. Paterson. Parametricity and dependent types. ICFP 2010: 345-356.

[4] P. Martin-Löf. An intuitionistic theory of types: predicative part. Logic Colloquium ’73 (Bristol, 1973), pp. 73118.

[5] P. Martin-Löf. About Models for Intuitionistic Type Theories and the Notion of Definitional Equality. Proceedings of the Third Scandinavian Logic Symposium, 1975, Pages 81-109.