An algorithm for testing conversion in Type Theory

Thierry Coquand

INRIA and University of Göteborg/Chalmers

Introduction

The goal of this note is to present a "modular" proof, for various type systems with η-conversion, of the completeness and correctness of an algorithm for testing the conversion of two terms. The proof of completeness is an application of the notion of logical relations (see Statman 1983 [14], that uses also this notion for a proof of Church-Rosser for simply typed λ-calculus).

An application of our result will be the equivalence between two formulations of Type Theory, the one where conversions are judgement, like in the present version of Martin-Löf set theory, and the one where conversion is defined at the level of raw terms, like in the standard version of LF (for a "conversion-as-judgement" presentation of LF, see Harper 1988 [6]). Even if we don't include η-conversion, the equivalence between the "conversion-as-judgement" and "conversion defined on raw terms" formulation appears to be a non-trivial property.

In order to simplify the presentation we will limit ourselves to type theory with only Π, and one universe. This calculus contains LF. After some motivations, we present the algorithm, the proof of its completeness and, as a corollary, its correctness. As a corollary of our argument, we prove normalisation, Church-Rosser, and the equivalence between the two possible formulations of Type Theory.

1 Informal motivation

1.1 The algorithm

The idea is to compute the weak head-normal form of the two terms (in an untyped way), and, in order to take care of η-conversion, in the case where

[1]This research was partly supported by ESPRIT Basic Research Action "Logical Frameworks".

one weak-head normal form is an abstraction $(\lambda x{:}\ A)M$ and the other is N a variable or an application, to compare recursively $apply(N, \xi)$ and $M[\xi]$.

This algorithm can be applied also for Authomath like system (and General Type Systems extended with η-conversion). But it is not complete if the type system does not have the normalisation property. It is directly used for type-checking and proof-checking in Type Theory.

1.2 Some remarks about the rules of Type Theory

The syntax is the following

$$M \ := \ \xi \mid U \mid apply(M, M) \mid (\lambda x{:}\ M)M \mid (\Pi x{:}\ M)M$$

We will denote by EXP the set of syntactic closed expressions. We make a distinction between free, or real variables, or parameters, written $\xi, \zeta, \ldots$, and the bound, or apparent, variables, written $x, y, z, \ldots$ If M is an expression with a bound variable x, and N any expression, we will denote by $M[N]$ the expression $[x = N]M$, which behaves like an ordinary substitution except that $[x = N]((\lambda y{:}\ A)M) = (\lambda y{:}\ A)M$ and $[x = N]((\Pi y{:}\ A)M) = (\Pi y{:}\ A)M$ if $y = x$. We denote by PAR the set of parameters, which is supposed to be infinite. We don't assume that terms are considered up to α-conversion. This is crucial if we want to describe really an actual implementation. This will be possible by an indexing over finite sets of parameters and later by an indexing over contexts.

Given a finite subset $I \subset PAR$, we denote by $EXP(I)$ the set of expressions whose free variables belong to I. If $\xi \in PAR$ does not belong to I, we denote by I, ξ the set $I \cup \{\xi\}$.

The rules of Type Theory are presented in the appendix. They describe inductively when a judgement J holds in a context Γ. There are four possible forms of judgement, that are A *set*, $A = B$, $a \in A$ and $a = b \in A$.

Let us define an order relation between context by $\Gamma \subseteq \Gamma_1$ iff if $\xi \in A$ is in Γ, then it is also in Γ_1. A direct inductive argument shows that if a judgement holds in Γ, and $\Gamma \subseteq \Gamma_1$, then the same judgement holds also in Γ_1. From now on, we will consider contexts as "possible worlds" in a Kripke-like manner. This is a convenient way of making precise the notion of parameters "available at a given moment of time" (see [5] for another example of this method).

Lemma 1 *If a judgement J holds in a context $\Gamma_1, \xi : A, \Gamma_2$, and $M \in A$ in the context Γ_1, then $(\xi/M)J$ holds in the context $\Gamma_1, (\xi/M)\Gamma_2$. If B set in the context $\Gamma_1, \xi : A, \Gamma_2$ and $M_1 = M_2 \in A$ in the context Γ_1 then $(\xi/M_1)B = (\xi/M_2)B$ in the context $\Gamma_1, (\xi/M_1)\Gamma_2$. If $N \in B$ in the context $\Gamma_1, \xi : A, \Gamma_2$ and $M_1 = M_2 \in A$ in the context Γ_1 then $(\xi/M_1)N = (\xi/M_2)N \in (\xi/M_1)B$ in the context $\Gamma_1, (\xi/M_1)\Gamma_2$.*

This is directly proved by induction.

In this approach, substitution is a meta-operation on terms. Another possible formulation of Type Theory is to take substitution as an explicit term forming operation.

Once the substitution lemma is proved, it is direct to prove that if $A = B$ holds in Γ, then both A *set* and B *set* holds in Γ, and that if $M = N \in A$ holds in Γ, then both $M \in A$ and $N \in A$ holds in Γ.

Since we have chosen a "Russell-like" formulation of universes (see Martin-Löf 84 [9] for an explanation of this terminology), there are some lemmas to be proved about equality judgements between sets.

Lemma 2 *The following properties hold:*

1. *if $A = B$ and $A \in U$ or $B \in U$, then both A, B are of type U and $A = B \in U$,*
2. *if $A = B$, then either A, B are both U, or $A = B \in U$ or A is $(\Pi x\text{:}\, A_0)A_1$, B is $(\Pi y\text{:}\, B_0)B_1$, and $A_0 = B_0$, $A_1[\xi] = B_1[\xi]$ $[\xi \in A_0]$,*
3. *if A set and A is not U or a product, then we have $A \in U$,*
4. *if $A = U$ $[\Gamma]$, or $U = A$ $[\Gamma]$, then A is U,*
5. *if $M \in A$ $[\Gamma]$ and M is a product, A is U.*

Proof: The first claim is proved directly by induction. The second claim is proved using the first. The last three claims are proved by a direct induction.

One property that does not seem directly provable is strengthening, which says that if a judgement J holds in the context $\Gamma_1, \xi : A, \Gamma_2$ and ξ does not occur in Γ_2 and J, then J holds in Γ_1, Γ_2. This property will be a consequence of our main proposition 1. Strengthening will be essential

in proving closure by η-reduction, and the equivalence between the present formulation of Type Theory and a formulation where conversion is defined at the level of raw terms.

Because of this, it is essential to formulate the rule Π-equality 2, which expresses the rule of η-conversion, as the equality $f = (\lambda x\colon A)apply(f, x) \in (\Pi x\colon A)B$, if $f \in (\Pi x\colon A)B$. Indeed, it does not seem possible to prove directly that if $(\lambda x\colon A)apply(f, x) \in (\Pi x\colon A)B$ holds and x does not appear free in f, then f is typable in the empty context.

Another property that does not seem directly provable is closure by β-reduction. This is a consequence of the fact that Π is one-to-one.

1.3 About the correctness and completeness proof

In this paper, we will show first the completeness, and then the correctness of the algorithm described above. We want to stress two features of this proof.

The first is that we work with a type system where equality between sets or between terms is a judgement. We have thus four kinds of judgement, namely A *set*, $A = B$, $a \in A$ and $a = b \in A$. This is to be contrasted with a presentation of type theory where equality is defined at the level of raw terms, and there are only two judgements, namely A *set*, and $a \in A$. This last version was the first one chosen by Martin-Löf, see for instance Martin-Löf 72 [8], and the first version appears in Martin-Löf 84 [9], or Harper 87 [6]. It is not at all clear that these two presentations are equivalent, even in the case where there is only β-conversion. Actually, when we work with the equality as judgement version of type theory, and we define the computation of the head-normal form $M \Rightarrow c$ in an untyped way, it is not clear that if A *set* and $A \Rightarrow B$, then $A = B$, or even B *set*, and that if $a \in A$ and $a \Rightarrow b$, then $a = b \in A$. A key lemma in proving this appears to be that Π is one-to-one, that is, if $(\Pi x\colon A_1)B_1 = (\Pi y\colon A_2)B_2$, then $A_1 = A_2$ and $B_1[\xi] = B_2[\xi]$ $[\xi \in A_1]$. This will be a corollary of our proof, as well as the equivalence between the two formulations of type theory.

The second is that our proof will be a syntactic reflection of the semantical proof of consistency described in Martin-Löf 84. What we are doing here is thus very close to the interpretation presented in Smith 84 [13], but for a non-extensional theory. To each set A, we will associate one predicate Φ_A defined on syntactic expressions, and one equivalence relation Δ_A on the set of expressions that satisfy Φ_A. We will show then that if $M \in A$, then $\Phi_A(M)$ and if $M = N \in A$, then $\Delta_A(M, N)$. We will also show that if

$\Phi_A(M)$ then M is normalisable, and if $\Delta_A(M,N)$ then M,N have a common β,η reduct. As corollary of the correctness proof of the algorithm, we will get the normalisation and the Church-Rosser property.

We can thus see our proof as a generalisation of the usual computability method, as in Martin-Löf 72. In this generalisation, one defines inductively a predicate and one equivalence relation on the set defined by this predicate, instead of defining only one predicate.

2 Weak head-normal form

We will say that a term is **canonical** if, and only if, it is U or an abstraction or a product.

The notion of weak head-normal form is given by its operational semantics.

$$\xi \Rightarrow \xi$$

$$U \Rightarrow U$$

$$(\lambda x\colon A)M \Rightarrow (\lambda x\colon A)M$$

$$(\Pi x\colon A)M \Rightarrow (\Pi x\colon A)M$$

$$\frac{M \Rightarrow (\lambda x\colon A)M_1 \quad M_1[N] \Rightarrow P}{apply(M,N) \Rightarrow P}$$

$$\frac{M \Rightarrow M_1}{apply(M,N) \Rightarrow apply(M_1,N)} \quad M_1 \textit{ not canonical}$$

We say that M has a **weak head-normal form** N iff $M \Rightarrow N$. M_1 and M_2 are **weakly equivalent**, notation $M_1 \simeq M_2$ iff M_1 and M_2 have identical weak head-normal forms. A term is **simple** iff it has a weak head-normal form which is not canonical. Notice that a weak head-normal form that is not canonical is either a parameter, or of the form $apply(N,M)$ where N is a weak head-normal form that is not canonical.

It is important to notice the difference between the relation $M_1 \simeq M_2$ and Kleene equality, which would be defined as: if M_1 (resp. M_2) has a weak head-normal form, then so has M_2 (resp. M_1) and they are identical. With the present definition, $M_1 \simeq M_2$ implies that both M_1 and M_2 have a weak head-normal form.

Lemma 3 *The following facts hold:*

1. *a given term has at most one weak head-normal form,*
2. *if $M \in EXP(I)$ and $M \Rightarrow N$, then $N \in EXP(I)$,*
3. *If $apply(M, N)$ has a weak head-normal form, then so does M,*
4. *if $M_1 \simeq M_2$, and $apply(M_1, N)$ has a weak head-normal form, then $apply(M_1, N) \simeq apply(M_2, N)$.*

Remark: The first claim says that the algorithm described by the relation $\Rightarrow$ is deterministic, The last claim is false in general if $apply(M_1, N)$ has no weak head-normal form. For instance, with $\Delta = (\lambda x\colon U)apply(x, x)$, we have that $\Delta \simeq \Delta$, but not that $apply(\Delta, \Delta) \simeq apply(\Delta, \Delta)$.

3 An algorithm for β, η-conversion

3.1 The algorithm

We define recursively when two terms M_1 and M_2 are "equivalent", notation $M_1 \Leftrightarrow M_2$. This will be defined between closed expressions, and we need to consider also the relation $M_1 \Leftrightarrow M_2$ $[I]$, I finite subset of PAR, $M_1, M_2 \in EXP(I)$. A consequence of the indexing by a finite set of parameters (and, later, of the use of contexts as Kripke world) is that we don't have to assume anything about α-conversion. The indexing by a finite set of parameters follows also the actual implementation of the algorithm, where we keep track of the real variables used so far in order to create a fresh variable.

$M \Leftrightarrow N$ $[I]$ if, and only if, M has a weak head-normal form M_0, N has a weak head-normal form N_0 and the pair (M_0, N_0) is of one of the following form:

- (ξ, ξ),
- (U, U),

- $((\lambda x\colon A_1)M_1, (\lambda y\colon A_2)N_1)$ and $M_1[\xi] \Leftrightarrow N_1[\xi]$ $[I, \xi]$,
- $((\Pi x\colon M_1)M_2, (\Pi y\colon N_1)N_2)$ with $M_1 \Leftrightarrow N_1$ $[I]$ and $M_2[\xi] \Leftrightarrow N_2[\xi]$ $[I, \xi]$,
- $(apply(M_1, M_2), apply(N_1, N_2))$ with $M_1 \Leftrightarrow N_1$ $[I]$ and $M_2 \Leftrightarrow N_2$ $[I]$,
- $((\lambda x\colon A)T, N_0)$ with $T[\xi] \Leftrightarrow apply(N_0, \xi)$ $[I, \xi]$, where N_0 is not canonical,
- $(M_0, (\lambda x\colon A)T)$ with $apply(M_0, \xi) \Leftrightarrow T[\xi]$ $[I, \xi]$, where M_0 is not canonical.

It can be shown that the choice of the "generic parameter" ξ such that ξ does not belong to I is irrelevant (there exists such a ξ because PAR is infinite). From this remark, we see that if $M_1 \Leftrightarrow M_2$ $[I]$ and $I \subseteq I_1$, then $M_1 \Leftrightarrow M_2$ $[I_1]$.

One of the goal of the paper is to show that, if M, N are two syntactic expressions that are sets, or are terms of the same type, then M, N are convertible iff $M \Leftrightarrow N$.

It is clear from this definition that $\Leftrightarrow$ is symmetric. Furthermore, if $M_1 \simeq M_2$ and $M_2 \Leftrightarrow N$, then $M_1 \Leftrightarrow N$.

Notice that the algorithm described by $\Leftrightarrow$ "forgets" the type of the abstractions. Intuitively, this is because $M \Leftrightarrow N$ is considered only if it is known already that M, N *set* or $M, N \in A$. This is also essential for the next two lemmas.

Lemma 4 *If $M_1, M_2 \in EXP(I)$ verify $apply(M_1, \xi) \Leftrightarrow apply(M_2, \xi)$ $[I, \xi]$, then $M_1 \Leftrightarrow M_2$ $[I]$. If $a \Leftrightarrow b$ $[I]$, and a, b are simple, then $apply(a, M) \Leftrightarrow apply(b, M)$ $[I_1]$ for any $I \subseteq I_1$, $M \in EXP(I_1)$.*

Proof: We suppose that $M_1, M_2 \in EXP(I)$ verify $apply(M_1, \xi) \Leftrightarrow apply(M_2, \xi)$ $[I, \xi]$. Then, both $apply(M_1, \xi)$ and $apply(M_2, \xi)$ have a weak head-normal form. It follows by lemma 3 that both M_1, M_2 have a weak head-normal form, that are abstraction or non-canonical. There are then four cases that can be checked directly. For instance, if $M_1 \Rightarrow (\lambda x : A)T$ and $M_2 \Rightarrow M_0$, M_0 not canonical, $apply(M_1, \xi) \Leftrightarrow apply(M_2, \xi)$ $[I, \xi]$ is equivalent to $T[\xi] \Leftrightarrow apply(M_0, \xi)$ from which $M_1 \Leftrightarrow M_2$ follows.

The other part is direct.

Lemma 5 $\Leftrightarrow$ *is a partial equivalence relation, i.e. is symmetric and transitive.*

Proof: By induction on the proof that $M_1 \Leftrightarrow M_2$, we show that if $M_2 \Leftrightarrow M_3$, then $M_1 \Leftrightarrow M_3$.

3.2 Normalisable terms

We define inductively a predicate $Norm$ on the set of syntactic expressions. It will be a family of predicates $Norm(M)$ $[I]$ defined on $EXP(I)$ such that $Norm(M)$ $[I]$ and $I \subseteq I_1$ implies $Norm(M)$ $[I_1]$. We say that $Norm(M)$ $[I]$, or that $M \in EXP(I)$ is **normalisable** iff $M \Rightarrow M_0$ and M_0 is of the form:

- ξ,
- U,
- $(\lambda x{:}\, M_1)M_2$ and $Norm(M_1)$ $[I]$, $Norm(M_2[\xi])$ $[I, \xi]$,
- $(\Pi x{:}\, M_1)M_2$ and $Norm(M_1)$ $[I]$, $Norm(M_2[\xi])$ $[I, \xi]$,
- $apply(M_1, M_2)$ and $Norm(M_1)$ $[I]$, $Norm(M_2)$ $[I]$.

Notice that given a proof of $Norm(M_1), Norm(M_2)$ $[I]$, we can decide whether or not $M_1 \Leftrightarrow M_2$ $[I]$, that is, $\Leftrightarrow$ is a decidable relation on the set of normalisable terms. Notice also that if $Norm(M)$ $[I]$, then $M \Leftrightarrow M$ $[I]$. The relation $Bisim$ is the equivalence relation on $\{M \in EXP \mid Norm(M)\}$ which is the restriction of $\Leftrightarrow$ on this set.

4 Computability relation

4.1 Contexts as Kripke possible world

We have four inductively defined relations on the set of syntactic expressions that correspond to the four judgement of type theory. They are described in the appendix. If Γ is a context, and I the finite set of parameters that occur in Γ, we will write $EXP(\Gamma)$, $M_1 \Leftrightarrow M_2$ $[\Gamma]$ and $Norm(M)$ $[\Gamma]$ respectively for $EXP(I)$, $M_1 \Leftrightarrow M_2$ $[I]$ and $Norm(M)$ $[I]$.

We recall that we have defined an order relation between context by $\Gamma \subseteq \Gamma_1$ iff if $\xi \in A$ is in Γ, then it is also in Γ_1. A direct inductive argument

shows that if a judgement holds in Γ, and $\Gamma \subseteq \Gamma_1$, then the same judgement holds also in Γ_1. From now on, we will consider contexts as "possible worlds" in a Kripke-like manner. This is a convenient way of making precise the notion of parameters "available at a given moment of time" (see [5] for another example of this method).

From now on, we take the convention that any statement or proof is, even if it is not stated explcitely, relativised to an arbitrary context.

For instance, a set X is now a family $X(\Gamma)$ of sets such that $X(\Gamma_1) \subseteq X(\Gamma_2)$ if $\Gamma_1 \subseteq \Gamma_2$. We will denote also by $x \in X$ $[\Gamma]$ the fact that x belongs to $X(\Gamma)$, and we will say in this case that x belongs to X **at level** Γ. A predicate φ on X is a now a proposition $\varphi(x)$ $[\Gamma]$ depending on a context Γ and on $x \in X(\Gamma)$, which is increasing in the context Γ, that is such that $\varphi(x)$ $[\Gamma]$ and $\Gamma \subseteq \Gamma_1$ implies $\varphi(x)$ $[\Gamma_1]$. There is a similar definition for relations. An example of such a set is the set of expressions. A predicate on this set is the predicate $Norm$, and a binary relation on this set is $\Leftrightarrow$.

Let φ_1, φ_2 be two predicates on EXP. Following the rules of Kripke semantics, we say that $\varphi_1(x)$ implies $\varphi_2(x)$ at level Γ iff for all $\Gamma_0 \supseteq \Gamma$, $\varphi_1(x)$ $[\Gamma_0]$ implies that $\varphi_2(x)$ $[\Gamma_0]$.

We let $X(\Gamma)$ be the set of pairs (ψ, δ) where ψ is a predicate on EXP at level Γ and δ an equivalence relation on $\{M \in EXP \mid \psi(M)\}$ at level Γ. The equality on $X(\Gamma)$ is $(\varphi_1, \delta_1) = (\varphi_2, \delta_2)$ $[\Gamma]$ iff $\varphi_1(M) \equiv \varphi_2(M)$ and $\delta_1(M, N) \equiv \delta_2(M, N)$ at level Γ (where $\equiv$ denotes logical equivalence).

4.2 Definition of the computability relation

If *A set*, let us say that a pair $(\Phi, \Delta) \in X$ is a **computability relation on** A iff the following conditions are satisfied:

1. $\Phi(a)$ implies $a \in A$,
2. $\Delta(a, b)$ implies $a = b \in A$,
3. $a \in A$, a simple and $Norm(a)$ imply $\Phi(a)$,
4. $a = b \in A$, a, b simple and $Bisim(a, b)$ imply $\Delta(a, b)$,
5. if $\Phi(a)$ then $Norm(a)$,

6. if $\Delta(a,b)$ then $a \Leftrightarrow b$,

7. if $\Phi(a)$, $a = u \in A$, $a \simeq u$, then $\Phi(u)$,

8. if $\Delta(a,b)$, $u = a \in A$, $u \simeq a$, $b = v \in A$, $b \simeq v$, then $\Delta(u,v)$.

We are going to define a predicate Ψ on EXP. Intuitively, $\Psi(A)$ means that A is a "well-formed" set. We will have that $\Psi(A)$ implies A *set*. For A such that $\Psi(A)$, we will define $\Theta_A = (\Phi_A, \Delta_A) \in X$ computability relation on A.

We first define Φ_U. Actually we define simultaneously Φ_U such that $\Phi_U(A)$ implies $A \in U$, and for A such that $\Phi_U(A)$ we define $\theta_A = (\varphi_A, \delta_A) \in X$ computability relation on A.

We say that $\Phi_U(A)$ [Γ] iff $A \in U$ [Γ] and

- either A is simple, in which case we ask $Norm(A)$ and we define $\varphi_A(M)$ [Γ_1], for $\Gamma_1 \supseteq \Gamma$, by $M \in A$, $Norm(M)$ at level Γ_1 and $\delta_A(M_1, M_2)$ is $M_1 = M_2 \in A$ & $M_1 \Leftrightarrow M_2$ at level Γ_1,

- or $A \Rightarrow (\Pi x\colon A_0)A_1$, in which case we ask

  1. $A = (\Pi x\colon A_0)A_1 \in U$,
  2. $\Phi_U(A_0)$,
  3. $\varphi_{A_0}(a)$ implies $\Phi_U(A_1[a])$,
  4. $\delta_{A_0}(a,b)$ implies $\theta_{A_1[a]} = \theta_{A_1[b]}$.

  These last conditions are stated at level Γ.

  In this case, we define $\varphi_A(M)$ [Γ_1] for $\Gamma_1 \supseteq \Gamma$ by $M \in A$ and

  – $\varphi_{A_0}(a)$ implies $\varphi_{A_1[a]}(apply(M,a))$,
  – $\delta_{A_0}(a,b)$ implies $\delta_{A_1[a]}(apply(M,a), apply(M,b))$,
  – if $M \Rightarrow (\lambda y\colon B_0)T$, then $Norm(B_0)$.

  $\delta_A(M_1, M_2)$ is the equivalence relation on $\{M \in EXP \mid \varphi_A(M)\}$ defined by

  $$M_1 = M_2 \in A \;\&\; [\varphi_{A_0}(a) \Rightarrow \delta_{A_1[a]}(apply(M_1,a), apply(M_2,a))].$$

  These last definitions are stated at level Γ_1.

Remark: A priori, it is not clear that if $A \in U$, and $A \Rightarrow B$, then A is simple or B is a product. It will follow from the fact that Π is one-to-one that A is simple or B is a product, that $B \in U$, and that $A = B$.

Let us explicit the definition of $\varphi_A(M)$ $[\Gamma_1]$ in the case where $A \Rightarrow (\Pi x: : A_0)A_1$. It means first that the judgement $M \in A$ holds in the context Γ_1. Next, if $\Gamma_1 \subseteq \Gamma_2$, then $\varphi_{A_0}(a)$ $[\Gamma_2]$ implies $\varphi_{A_1}(apply(M,a))$ $[\Gamma_2]$ and $\delta_{A_0}(a,b)$ $[\Gamma_2]$ implies $\delta_{A_1[a]}(apply(M,a), apply(M,b))$ $[\Gamma_2]$. Finally, if $M \Rightarrow (\lambda y: B_0)T$ $[\Gamma_1]$ then $Norm(B_0)$ $[\Gamma_1]$.

Notice that if follows directly from this definition that $\Phi_U(A)$, $A = B \in U$, and $A \simeq B$ imply $\Phi_U(B)$ and $\theta_A = \theta_B$.

There is a problem a priori in such an inductive definition, because of the fact that the predicate Φ_U that is defined has negative occurences. A discussion on this difficulty is postponed to the next section. This discussion will justify also the following induction principle. Let P be a predicate on EXP. Suppose that if A is simple, then $\Phi_U(A)$ implies $P(A)$, and that if $\Phi_U(A)$, $A \Rightarrow (\Pi x: A_0)A_1$, $A = (\Pi x: A_0)A_1 \in U$, $P(A_0)$, $\varphi_{A_0}(a) \Rightarrow P(A_1[a])$, then $P(A)$. Then, we can conclude that, for all A, $\Phi_U(A)$ implies $P(A)$.

Lemma 6 *If $\Phi_U(A)$, then $Norm(A)$ and $\theta_A = (\varphi_A, \delta_A)$ is a computability relation on A.*

Proof: Remark that this entails that if $\Phi_U(A)$, then $\varphi_A(\xi)$ $[\xi \in A]$, because ξ is simple and normalisable.

Consider the property $P(A)$ that $\Phi_U(A)$ and

1. $Norm(A)$,
2. $a \in A$, a simple and $Norm(a)$ imply $\varphi_A(a)$,
3. $a = b \in A$, a, b simple and $Bisim(a,b)$ imply $\delta_A(a,b)$,
4. if $\varphi_A(a)$ then $Norm(a)$,
5. if $\delta_A(a,b)$ then $a \Leftrightarrow b$,
6. if $\varphi_A(a)$, $a = u \in A$, $a \simeq u$, then $\varphi_A(u)$,
7. if $\delta_A(a,b)$, $u = a \in A$, $u \simeq a$, $b = v \in A$, $b \simeq v$, then $\delta_A(u,v)$.

By definition of Φ_U, $P(A)$ if A is simple and $\Phi_U(A)$.

Next, suppose $\Phi_U(A)$, $A \Rightarrow (\Pi x\colon A_0)A_1$, $A = (\Pi x\colon A_0)A_1 \in U$, $P(A_0)$, and $\varphi_{A_0}(a)$ implies $P(A_1[a])$.

Notice first that, by induction hypothesis, $\varphi_{A_0}(\xi)$ $[\xi \in A_0]$, since ξ is simple and normalisable of type A_0 at level $\xi \in A_0$.

Since $\Phi_U(A)$, we have $\Phi_U(A_1[\xi])$ $[\xi \in A_0]$ and hence, $Norm(A_1[\xi])$ $[\xi \in A_0]$ by induction hypothesis. We have also $Norm(A_0)$ by induction hypothesis, and hence $Norm(A)$.

Let $M \in A$ be simple and normalisable. For any a such that $\varphi_{A_0}(a)$, we have that $apply(M, a)$ is simple and normalisable, because M is simple and a normalisable, and hence by induction hypothesis, $\varphi_{A_1[a]}(apply(M, a))$. For any a, b such that $\delta_{A_0}(a, b)$, we have that $apply(M, a)$ and $apply(M, b)$ are simple and that $Bisim(apply(M, a), apply(M, b))$, because $Bisim(a, b)$ and M is simple. Hence, by induction hypothesis, $\delta_{A_1[a]}(apply(M, a), apply(M, b))$. This shows that $\varphi_A(M)$ holds.

In the same way, we can show that if $M, N \in A$, M, N are simple and $Bisim(M, N)$, then $\delta_A(M, N)$, using lemma 4.

If $\varphi_A(M)$, then we have $\varphi_{A_1[\xi]}(apply(M, \xi))$, because $\varphi_{A_0}(\xi)$ $[\xi \in A_0]$. Hence $Norm(apply(M, \xi))$ at level $\xi \in A_0$. This implies that M is simple or has a weak head-normal form which is an abstraction. If M is simple, then $Norm(apply(M, \xi))$ implies $Norm(M)$. If $M \Rightarrow (\lambda y\colon B_0)T$, then $Norm(B_0)$. This, together with $Norm(apply(M, \xi))$, implies that M is normalisable.

If $\delta_A(M, N)$, then we have $\delta_{A_1[\xi]}(apply(M, \xi), apply(N, \xi))$, because we have $\varphi_{A_0}(\xi)$ $[\xi \in A_0]$. Hence $apply(M, \xi) \Leftrightarrow apply(N, \xi)$ at level $\xi \in A_0$ by induction hypothesis. We deduce $M \Leftrightarrow N$ by lemma 4.

Next, suppose that $a \in A$, $\varphi_A(a)$, $a = u \in A$, and $a \simeq u$ and we have to show that $\varphi_A(u)$ holds. We have $u \in A$ since $a = u \in A$. If $M \in A_0$ and $\varphi_{A_0}(M)$, then we have to show that $\varphi_{A_1[M]}(apply(u, M))$ holds. We have $\varphi_{A_1[M]}(apply(a, M))$ because $\varphi_A(a)$. Furthermore $apply(a, M) = apply(u, M) \in A_1[M]$. By lemma 8, we have that $apply(a, M)$ is normalisable and hence has a weak head-normal form. By lemma 3, we have $apply(a, M) \simeq apply(u, M)$. By induction hypothesis, the lemma holds for $A_1[M]$. Hence $\varphi_{A_1[M]}(apply(u, M))$. Furthermore, if $u \Rightarrow (\lambda x\colon B_0)T$, then $Norm(B_0)$ because $u \simeq a$ and $\varphi_A(a)$. The proof of the last claim is similar.

Notice the essential use of contexts as Kripke worlds in this reasoning.

We define the equivalence relation Δ_U on the set of expressions that satisfies Φ_U by: $\Delta_U(A, B)$ iff $A = B \in U$, $\theta_A = \theta_B$, $A \Leftrightarrow B$ and if $A \Rightarrow (\Pi x\colon A_0)A_1$, $B \Rightarrow (\Pi y\colon B_0)B_1$ then $A_0 = A_1$ and $B_0[\xi] = B_1[\xi]$ $[\xi \in A_0]$.

Lemma 7 *(Φ_U, Δ_U) is a computability relation on U.*

We can now define $\Psi(A)$, and for A such that $\Psi(A)$, the predicate Φ_A and the equivalence relation Δ_A. We say that $\Psi(A)$ $[\Gamma]$ iff A *set* $[\Gamma]$ and

- either A is U, we have already defined Φ_U and Δ_U,
- or $A \in U$ $[\Gamma]$ and $\Phi_U(A)$ $[\Gamma]$, in which case we take Θ_A to be θ_A,
- or A is $(\Pi x\colon A_0)A_1$, in which case we ask
  - $\Psi(A_0)$,
  - $\Phi_{A_0}(a)$ implies $\Psi(A_1[a])$,
  - $\Delta_{A_0}(a,b)$ implies $\Theta_{A_1[a]} = \Theta_{A_1[b]}$.

  These last conditions are stated at level Γ.

  In this case, we define $\Phi_A(M)$ $[\Gamma_1]$ for $\Gamma_1 \supseteq \Gamma$ by $M \in A$ and

  - $\Phi_{A_0}(a)$ implies $\Phi_{A_1[a]}(apply(M,a))$,
  - $\Delta_{A_0}(a,b)$ implies $\Delta_{A_1[a]}(apply(M,a), apply(M,b))$,
  - if $M \Rightarrow (\lambda y\colon B_0)T$, then $Norm(B_0)$.

  $\Delta_A(M_1,M_2)$ is the equivalence relation on $\{M \in EXP \mid \varphi_A(M)\}$ defined by

  $$M_1 = M_2 \in A \;\&\; [\Phi_{A_0}(a) \Rightarrow \Delta_{A_1[a]}(apply(M_1,a), apply(M_2,a))].$$

  These last definitions are stated at level Γ_1.

Notice that this definition is a priori ambiguous, since we can have both $A \in U$, and A is $(\Pi x\colon A_0)A_1$. But in this case, we have also $A_0 \in U$, and $A_1[\xi] \in U$ $[\xi \in A_0]$, and we can show inductively on the proof of $A \in U$ that both cases give the same definition. This ambiguity does not appear if we use a formulation "à la Tarski" of universes, as in Martin-Löf 84 [9], or if we restrict the proof to LF, where there is a syntactic distinction between types and kinds.

Lemma 8 *If $\Psi(A)$, then $Norm(A)$ and $\Theta_A = (\Phi_A, \Delta_A)$ is a computability relation on A.*

Proof: The argument is the same as the one for Φ_U given above, and is by induction on the proof that $\Psi(A)$. We use furthermore the fact that the statement holds for U, using lemma 7 and 6.

4.3 Justification of this definition

The inductive definitions of Ψ, and of Φ_U are of the following form: we define simultaneously both a predicate φ on the set EXP, and a function on $\{M \in EXP \mid \varphi(M)\}$. For Φ_U for instance, we define simultaneously the predicate Φ_U and $\theta_A \in X$ defined for $A \in \{M \in EXP \mid \Phi_U(M)\}$. We have to convince ourselves that the above definition is correct. We will show how to reduce this kind of definition to the existence of a least fixed-point of a monotone operator on a complete lattice.

For a first such reduction, consider the set Y of pairs (φ, f) where φ is a predicate on EXP and f a function from $\{M \in EXP \mid \varphi(M)\}$ to X. Notice first that Y is a complete lattice for the ordering $(\varphi_1, f_1) \leq (\varphi_2, f_2)$ defined by $\varphi_1(M) \Rightarrow \varphi_2(M)$ and, if $\varphi_1(M)$, then $f_1(M) = f_2(M)$. Notice next that the definition of Φ_U can be seen as a monotone operator from Y to Y. This is essentially the justification implicit in Martin-Löf 72 [8] which is explained in another framework in Aczel 77 [1].

For a second reduction, more set-theoretical in nature, we consider $A \longmapsto (\varphi_A, \delta_A)$ as a functional relation R between EXP and X, that is a relation R on $EXP \times X$ such that, for any $M \in EXP$, there exists at most one $x \in X$ such that $R(M, x)$. Notice first that the definition of Φ_U can be seen as a monotone operator on the set of relation between EXP and X. This is a complete lattice for the inclusion. Notice next that the least fixed-point of this operator is a functional relation. We define then Φ_U to be the domain of this functional relation, that is the set of $M \in EXP$ such that there exists $x \in X$ such that $R(M, x)$. This reduction appears in Allen 87 [2], and was pointed out to the author by S. Hayashi.

Of course, we have now to show that a monotone function on a complete lattice has a least fixed-point, at least in these particular cases. It is well-known how to do it using an impredicative definition (this is Tarski fixed-point theorem). One may wonder if there exist more basic reductions. One alternative is even to admit the existence of Φ_U and the induction principle over it as a new axiom. We will limit ourselves here to have indicated these two possible reductions to the existence of a fixed-point of a monotone function on a complete lattice.

5 Completeness of the algorithm

We define first inductively when a "type-checking" context $\Gamma = \xi_1 \in A_1, \ldots, \xi_n \in A_n$ is valid, and when it is valid, when a substitution, written $(\xi_1/a_1)\ldots(\xi_n/a_n)$, fits this context at level Γ_0, and when two such substitutions $(\xi_1/a_1)\ldots(\xi_n/a_n)$ and $(\xi_1/b_1)\ldots(\xi_n/b_n)$ are considered to be equal at level Γ_0.

If $\Gamma = \xi \in A$, then Γ is valid iff $\Psi(A)$. Furthermore (ξ/a) fits Γ at level Γ_0 iff $\Phi_A(a)$ $[\Gamma_0]$ and $(\xi/a), (\xi/b)$ are equal iff $\Delta_A(a,b)$ $[\Gamma_0]$.

Next, if Γ is valid, if A *set* $[\Gamma]$, and $\Psi(\sigma A)$ for any σ that fits Γ at level Γ_0, and $\Theta_{\sigma_1 A} = \Theta_{\sigma_2 A}$ $[\Gamma_0]$ whenever σ_1 and σ_2 are equal, then $\Gamma, \xi \in A$ is valid. Furthermore, $\sigma(\xi/a)$ fits $\Gamma, \xi \in A$ at level Γ_0 iff $\Phi_{\sigma A}(a)$ $[\Gamma_0]$, and $\sigma_1(\xi/a_1), \sigma_2(\xi/a_2)$ are equal iff σ_1, σ_2 are equal and $\Delta_{\sigma_1 A}(a_1, a_2)$ $[\Gamma_0]$.

Proposition 1 *If A set, then $\Psi(A)$. If $A = B$, then $\Psi(A)$, $\Psi(B)$, $\Theta_A = \Theta_B$. If $a \in A$, then $\Psi(A)$ and $\Phi_A(a)$. Finally, if $a = b \in A$, then $\Psi(A)$ and $\Delta_A(a,b)$.*

Proof: More generally, we prove inductively that if Γ is a valid context, σ a substitution that fits Γ at level Γ_0, and σ_1, σ_2 two equal substitutions that fit Γ at level Γ_0, then

- if A *set* $[\Gamma]$, then, at level Γ_0, $\Psi(\sigma A)$, and $\Psi(\sigma_1 A)$, $\Psi(\sigma_2 A)$, and $\Theta_{\sigma_1 A} = \Theta_{\sigma_2 A}$,
- if $A = B$ $[\Gamma]$, then, at level Γ_0, $\Psi(\sigma A)$, $\Psi(\sigma B)$ and $\Theta_{\sigma A} = \Theta_{\sigma B}$,
- if $a \in A$ $[\Gamma]$, then, at level Γ_0, $\Psi(\sigma A)$, $\Phi_{\sigma A}(\sigma a)$, and $\Theta_{\sigma_1 A} = \Theta_{\sigma_2 A}$, $\Delta_{\sigma_1 A}(\sigma_1 a, \sigma_2 a)$,
- if $a = b \in A$ $[\Gamma]$, then, at level Γ_0 $\Psi(\sigma A)$, $\Theta_{\sigma_1 A} = \Theta_{\sigma_2 A}$, and $\Delta_{\sigma A}(\sigma a, \sigma b)$.

This is proved by induction together with the fact that any context is valid. Lemma 8 handles the rules of β-conversion and the rule of η-conversion.

Let us show for instance how is handled the rule Π-equality 2. To simplify the notations, we suppose the context empty. We have then $\Psi((\Pi x{:}\, A)B), \Phi_{(\Pi x A)B}(f)$, and we want to show $\Phi_{(\Pi x A)B}((\lambda x{:}\, A)apply(f,x))$ and $\Delta_{(\Pi x A)B}(f, (\lambda x{:}\, A)apply(f,x))$. By the definition of Ψ, we have that $\Psi(A)$, that $\Phi_A(a)$ implies $\Psi(B[a])$, and $\Delta_A(a,b)$ implies $\Theta_{B[a]} = \Theta_{B[b]}$, $\Phi_A(a)$ implies $\Phi_{B[a]}(apply(f,a))$ and $\Delta_A(a,b)$ implies $\Delta_{B[a]}(apply(f,a), apply(f,b))$. Let us assume $\Phi_A(a)$ and show that $\Phi_{B[a]}(apply((\lambda x{:}\, : A)apply(f,x), a))$. This follows from $\Phi_{B[a]}(apply(f,a))$ and lemma 8. Since $Norm(A_0)$ because $\Psi(A_0)$ and by lemma 8, we have $\Phi_{(\Pi x A)B}((\lambda x{:}\, A)apply(f,x))$. In the same way, if $\Delta_A(a,b)$, then we have that $\Delta_{B[a]}(apply((\lambda x{:}\, A)apply(f,x), a), apply((\lambda x{:}\, A)apply(f,x), b))$ follows from the fact that we have $\Delta_{B[a]}(apply(f,a), apply(f,b))$ and lemma 8.

In particular, if $M = N$, or $M = N \in A$, then $M \Leftrightarrow N$ and this expresses the completeness of our algorithm for testing conversion of terms.

6 Correctness of the algorithm

A first application is the fact that Π is one-to-one.

Proposition 2 *If* $(\Pi x\colon A_0)A_1 = (\Pi y\colon B_0)B_1$, *then* $A_0 = B_0$ *and* $A_1[\xi] = B_1[\xi]\ [\xi \in A_0]$.

Proof: We use lemma 2. The result is clear in the case of Π-formation 2. And if $(\Pi x\colon A_0)A_1 = (\Pi y\colon B_0)B_1 \in U$, then $\Delta_U((\Pi x\colon A_0)A_1, (\Pi y\colon B_0)B_1)$ by proposition 1, hence the result.

The rest of this section collects direct consequences of the fact that Π is one-to-one.

Corollary 1 *If* $M \in A_1$, $M \in A_2$, *then* $A_1 = A_2$.

Proof: By induction on M using proposition 2.

Lemma 9 *If* $(\lambda x\colon A_1)b \in (\Pi y\colon A_2)B$, *then* $A_1 = A_2$ *and* $b[\xi] \in B[\xi]\ [\xi \in A_1]$.

Proof: We have $(\lambda x\colon A_1)b \in (\Pi x\colon A_1)B_1$, with $b[\xi] \in B_1[\xi]\ [\xi \in A_1]$ and $(\Pi x\colon A_1)B_1 = (\Pi y\colon A_2)B$. By proposition 2, we get $A_1 = A_2$ and $B_1[\xi] = B[\xi]\ [\xi \in A_1]$, hence he result.

Lemma 10 *If* A *set and* $A \Rightarrow B$, *then* B *set and* $A = B$. *If* $a \in A$, *and* $a \Rightarrow b$, *then* $b \in A$ *and* $a = b \in A$.

Proof: We show by induction on $M \Rightarrow N$, that if $M \Rightarrow N$ then if M *set*, then N *set* and $M = N$, and if $M \in A$, then $N \in A$ and $M = N \in A$.

Let us show the case where M is $apply(M_1, N_1)$, $M_1 \Rightarrow (\lambda x\colon A_2)M_2$ and $M_2[N_1] \Rightarrow N$. We have $M_1 \in (\Pi x\colon A_1)B_1$, $N_1 \in A_1$. By induction hypothesis, $(\lambda x\colon A_2)M_2 \in (\Pi x\colon A_1)B_1$ and $M_1 = (\lambda x\colon A_2)M_2 \in (\Pi x\colon A_1)B_1$. By lemma 9, $A_1 = A_2$ and $M_2[\xi] \in B_1[\xi]\ [\xi \in A_1]$. We deduce that $M_2[N_1] \in B_1[N_1]$. By induction hypothesis, $N \in B_1[N_1]$ and $M_2[N_1] = N \in B_1[N_1]$. But we have $apply(M_1, N_1) = apply((\lambda x\colon A_2)M_2, N_1) \in B_1[N_1]$ by Π-elimination 2, and $apply((\lambda x\colon A_2)M_2, N_1) = M_2[N_1] \in B_1[N_1]$ by Π-equality 1.

This lemma can be expressed as the statement of the subject-reduction for β-reduction.

The next proposition states the correctness of the algorithm corresponding to the relation $\Leftrightarrow$. This is only stated in the empty context, but the relativised version to any context holds as well.

Proposition 3 *If A set, B set, and $A \Leftrightarrow B$ then $A = B$. Similarly, if $a \in A$, $b \in A$ and $a \Leftrightarrow b$, then $a = b \in A$. Furthermore, if $a \Leftrightarrow b$, $a \in A$, $b \in B$, and a, b are simple, then $A = B$, and $a = b \in A$.*

Proof: We use essentially lemma 10 and lemma 2. We prove simultaneously by induction on $M \Leftrightarrow N$ that if $M \Leftrightarrow N$, then if M, N *set* then $M = N$, if $M, N \in A$ then $M = N \in A$, and if M, N are simple and $M \in A, N \in B$, then $A = B$ and $M = N \in A$.

Let us consider for instance the case where $M \Rightarrow apply(M_1, M_2)$ and $N \Rightarrow apply(N_1, N_2)$ and $M_1 \Leftrightarrow M_2$, $N_1 \Leftrightarrow N_2$. We have that M, N are simple. If $M \in A$, $N \in B$, then we have $M_1 \in (\Pi x{:}\, C)A_1$, $M_2 \in C$, $N_1 \in (\Pi y{:}\, D)B_1$, $N_2 \in D$ and $A_1[M_2] = A$, $B_1[N_2] = B$. This follows from lemma 10. Since M_1, N_1 are simple, we can apply the induction hypothesis and we get that $(\Pi x{:}\, C)A_1 = (\Pi y{:}\, D)B_1$, and $M_1 = N_1 \in (\Pi x{:}\, C)A_1$. By proposition 2, this implies $C = D$ and $A_1[\xi] = B_1[\xi]$ $[\xi \in C]$. We have then $M_2, N_2 \in C$. By induction hypothesis, this implies $M_2 = N_2 \in C$. We then get that $apply(M_1, M_2) = apply(N_1, N_2) \in A_1[M_2]$, and $A = A_1[M_2] = B_1[N_2] = B$.

Corollary 2 *If A, B set and $A = B$ $[\Gamma]$, then $A = B$. If $M, N \in A$ and $M = N \in A$ $[\Gamma]$, then $M = N \in A$.*

Proof: By proposition 1 and proposition 3.

The relativised version of this corollary says that the equational theory at level Γ_1 is a conservative extension of the one at level $\Gamma \subseteq \Gamma_1$.

7 Equivalence with another formulation of Type Theory

Lemma 11 *If M set then M has a β-normal form M_0 such that $M = M_0$. If $M \in A$, then M has a β-normal form M_0 such that $M = M_0 \in A$.*

Proof: We have $Norm(M)$ if M *set* or $M \in A$ by proposition 1. We prove next by induction on the proof that $Norm(M)$ using lemma 10 that if $Norm(M)$, then if M *set* then M has a normal form M_0 such that $M = M_0$, and if $M \in A$ then M has a normal form M_0 such that $M = M_0 \in A$.

Let us say that two expressions M, N are confluent if they can be reduced to a same term by β, η-reductions.

Proposition 4 *If $M = N$, or $M = N \in A$, then M, N are confluent.*

Proof: We define the size $s(M)$ of a term M as the number of symbols in M. We prove by induction on $s(M) + s(N)$ that if $M \Leftrightarrow N$, M, N in β-normal form then if M, N *set* then M, N are confluent, if $M = N \in A$ then M, N are confluent, and finally, if M, N are simple and $M \in A, N \in B$, then $A = B$ and M, N are confluent.

Let us treat only two cases. Let us suppose that M is $(\lambda x\colon P)M_1$ and N is $(\lambda y\colon Q)N_1$, and $M_1[\xi] \Leftrightarrow N_1[\xi]$, and $M = N \in A$. We have then $M_1[\xi] \in B_1[\xi]$ $[\xi \in P]$ and $N_1[\xi] \in C_1[\xi]$ $[\xi \in Q]$ with $A = (\Pi x\colon P)B_1 = (\Pi y\colon Q)C_1$. By proposition 2, we have $P = Q$, and $B_1[\xi] = C_1[\xi]$ $[\xi \in P]$, and thus $M_1[\xi] = N_1[\xi] \in B_1[\xi]$ $[\xi \in P]$. By induction hypothesis, we have that $M_1[\xi]$ and $N_1[\xi]$ are confluent. Furthermore, $P = Q$ implies $P \Leftrightarrow Q$ by proposition 1. By induction hypothesis, P and Q are confluent. Hence, M and N are confluent.

If M is simple, and N is $(\lambda x\colon T)N_1$, and $M = N \in A$, then we have $A = (\Pi x\colon T)B$ with $N_1[\xi] \in B[\xi]$ $[\xi \in T]$. We have then $apply(M, \xi) = N_1[\xi] \in B[\xi]$ $[\xi \in T]$. By induction hypothesis, using the fact that $s(apply(M, \xi)) + s(N_1[\xi]) < s(M) + s(N)$, we get that $apply(M, \xi)$ and $N_1[\xi]$ are confluent. Hence, M and N are confluent.

We can then apply proposition 1, and lemma 11.

Lemma 12 *If A set and $A = (\Pi x\colon B_0)B_1$ $[\Gamma]$, then $A = (\Pi x\colon A_0)A_1$ with $A_0 = B_0$ $[\Gamma]$ and $A_1[\xi] = B_1[\xi]$ $[\Gamma, \xi \in A_0]$.*

Proof: This is clear if $A = (\Pi x\colon B_0)B_1$ $[\Gamma]$ can be derived by Π-formation 2. If $A = (\Pi x\colon B_0)B_1 \in U$ $[\Gamma]$, then $\Delta_U(A, (\Pi x\colon B_0)B_1)$ at level Γ, and so $A \Rightarrow (\Pi x\colon A_0)A_1$, and at level Γ, $A = (\Pi x\colon A_0)A_1 \in U$, $A_0 = A_1 \in U$, $B_0[\xi] = B_1[\xi] \in U$ $[\xi \in A_0]$. By lemma 10, we get actually that $A = (\Pi x\colon A_0)A_1$ in the empty context.

Lemma 13 *If A set $[\Gamma_1, \xi : B, \Gamma_2]$ and ξ does not appear in Γ_2 and A, then A set $[\Gamma_1, \Gamma_2]$. If $M \in A$ $[\Gamma_1, \xi : B, \Gamma_2]$ and ξ does not appear in Γ_2 and M, then there exists A' set $[\Gamma_1, \Gamma_2]$ such that $M \in A'$ $[\Gamma_1, \Gamma_2]$ and $A = A'$ $[\Gamma_1, \xi : B, \Gamma_2]$.*

Proof: By induction, using lemma 12 and corollary 2.

Corollary 3 *If the judgement J holds in the context $\Gamma_1, \xi : A, \Gamma_2$ and ξ does not appear in Γ_2 and J, then J holds in Γ_1, Γ_2.*

Notice that this lemma does not hold in extensional Type Theory (like the one of Martin-Löf 84 [9]).

Lemma 14 *The subject reduction property holds for η-reduction.*

Proof: This means that, if $(\lambda x : A)apply(f, x) \in C$ and x is not free in f, then $f = (\lambda x\colon A)apply(f, x) \in C$.

Indeed, we have $C = (\Pi x\colon A)B$, with $apply(f, \xi) \in B[\xi]$ $[\xi \in A]$. Hence, at level $\xi \in A$, the type of f is a product $(\Pi x\colon A_1)B_1$ and we have that $A = A_1$, and $B_1[\xi] = B[\xi]$. By lemma 13 and lemma 12, we deduce that $f \in (\Pi x\colon A_2)B_2$ in the empty context, with, at level $\xi \in A$, $A_1 = A_2$ and $B_2[\zeta] = B_1[\zeta]$ $[\zeta \in A_2]$. This means $A_1 = A_2$ $[\xi \in A]$ and $B_1[\zeta] = B_2[\zeta]$ $[\xi \in A, \zeta \in A_2]$.

This implies $B_2[\xi] = B[\xi]$ $[\xi \in A]$. By corollary 2, we have also $A_2 = A$ in the empty context. Hence, $(\Pi x\colon A)B = (\Pi x\colon A_2)B_2$ and f is of type $(\Pi x\colon A)B = C$. By the rule of Π-equality 2, and by the rule of Set equality, we get $f = (\lambda x\colon\colon A)apply(f, x) \in C$.

Proposition 5 *If A, B set and A, B are confluent, then $A = B$.*

It is now clear the "conversion-as-judgements" version of type theory is equivalent to the version where conversion is defined at the level of raw terms, like for the usual presentation of LF [6] or in [8]. Indeed, one can see a priori that if a judgement holds for the "conversion-as-judgment" version, it holds for the other version. Proposition 4 shows the converse.

One can also deduce the decidability of type-checking, following an usual argument (see for instance [5]).

Conclusion

We tried to present a direct, semantically motivated, proof of the correctness and completness of an algorithm that tests conversion in type theory. Our proof can be seen as an expression of one possible semantics of type theory. It applies also to Edinburgh LF. It may be interesting to apply it for the case of set theory expressed in Martin-Löf's logical framework.

Acknowledgement and related works

In [7], D. Howe proves the fact that Π is one-to-one for NuPrl and extensional Type Theory.

In [12], A. Salvesen proves that Π is one-to-one and the Church-Rosser property for the version of LF where "conversion defined on raw terms".

The problem of conversion of terms in presence of η-conversion is also studied in [4].

In [11], the idea of contexts as Kripke worlds is used for giving a constructive version of the notion of possible worlds.

I want to thank Catarina Svensson for pointing out to me that the equivalence between the two formulations of Type Theory was a non-trivial property. Thanks also to Bengt Nordström, Jan Smith, Lena Magnusson and Anne Salvesen for interesting discussions on this topic. Finally, I want to thank Per Martin-Löf for his remarks on a previous version of this paper.

A The Rules.

General rules.

Context formation

$$\frac{A\ set}{\xi \in A\ context}$$

$$\frac{\Gamma\ context}{\xi \in A\ [\Gamma]}$$

Where $\xi \in A$ in Γ.

$$\frac{\Gamma\ context \qquad A\ set\ [\Gamma]}{\Gamma, \xi \in A\ context}$$

Where ξ not in Γ.

The rules below are also valid when relativised to an arbitrary context. The restriction on the parameter ξ is that it is "generic" w.r.t. the conclusion of the rule, i.e. does not appear in this conclusion. It can be proved that its choice is irrelevant.

Reflexivity

$$\frac{a \in A}{a = a \in A} \qquad \frac{A\ set}{A = A}$$

Symmetry

$$\frac{a = b \in A}{b = a \in A} \qquad \frac{A = B}{B = A}$$

Transitivity

$$\frac{a = b \in A \qquad b = c \in A}{a = c \in A} \qquad \frac{A = B \qquad B = C}{A = C}$$

Set equality

$$\frac{a \in A \qquad A = B}{a \in B} \qquad \frac{a = b \in A \qquad A = B}{a = b \in B}$$

Cartesian Product of a Family of Sets.

Π – formation 1

$$\frac{A\ set \qquad B[\xi]\ set\ [\xi \in A]}{(\Pi x{:}\ A)B\ set}$$

Π – formation 2

$$\frac{A = C \qquad B[\xi] = D[\xi]\ [\xi \in A]}{(\Pi x{:}\ A)B = (\Pi y{:}\ C)D}$$

Π – introduction 1

$$\frac{b[\xi] \in B[\xi]\ [\xi \in A]}{(\lambda x{:}\ A)b \in (\Pi x{:}\ A)B}$$

Π – introduction 2

$$\frac{A_1 = A_2 \qquad b_1[\xi] = b_2[\xi] \in B[\xi]\ [\xi \in A_1]}{(\lambda x{:}\ A_1)b_1 = (\lambda x{:}\ A_2)b_2 \in (\Pi x{:}\ A_1)B}$$

Π – elimination 1

$$\frac{f \in \Pi(A, B) \qquad a \in A}{apply(f, a) \in B[a]}$$

Π – elimination 2

$$\frac{f = g \in (\Pi x\colon A)B \quad a = b \in A}{apply(f, a) = apply(g, b) \in B[a]}$$

Π – equality 1

$$\frac{b[\xi] \in B[\xi]\ [\xi \in A] \quad a \in A}{apply((\lambda x\colon A)b, a) = b[a] \in B[a]}$$

Π – equality 2

$$\frac{f \in (\Pi x\colon A)B}{f = (\lambda x\colon A)apply(f, x) \in (\Pi x\colon A)B}$$

The Set of Small Sets.

U – formation:

$$U\ set$$

U – introduction 1:

$$\frac{A \in U \qquad B[\xi] \in U\ [\xi \in A]}{(\Pi x\colon A)B \in U}$$

U – introduction 2:

$$\frac{A = C \in U \qquad B[\xi] = D[\xi] \in U\ [\xi \in A]}{(\Pi x\colon A)B = (\Pi y\colon C)D \in U}$$

Set – formation 1

$$\frac{A \in U}{A\ set}$$

Set – formation 2

$$\frac{A = B \in U}{A = B}$$

References

[1] Aczel. P. (1979), Frege structures and the notions of propositions, truth, and set in: Barwise, J., Keisler, H.J., and Kunene, K. (eds), *Logic Colloquium 77*, North-Holland, Amsterdam.

[2] Allen. S. (1987), A Non Type-Theoretic Semantics for Type-Theoretic Language, Ph. D. Thesis, Cornell U.

[3] Beeson M.J. (1984), *Foundations of Constructive Mathematics*, Springer-Verlag, Berlin.

[4] Breazu-Tannen V., Gallier J. (1989), Polymorphic rewriting conserves algebraic strong normalisation and confluence, Proceedings of ICALP, Stresa.

[5] Coquand Th., Gallier J. (1990), A Proof of Strong Normalisation Using a Kripke-Like Interpretation, Draft, in the proceeding of the first workshop on Logical Framework.

[6] Harper. R. (1988), An Equational Formulation of LF, LFCR Report Series, ECS-LFCS-88-67, Edinburgh.

[7] Howe. D. (1989), Equality In Lazy Computation Systems, in the proceedings of the fourth Logic in Computer Science.

[8] Martin-Löf. P. (1972), An Intuitionistic Theory of Types, Unpublished manuscript.

[9] Martin-Löf. P. (1984), *Intuitionistic Type Theory*, Studies in Proof Theory, Lecture Notes, Bibliopolis.

[10] Nordström B., Petersson K., Smith. J. M. (1990), *Programming in Martin-Löf Type Theory*, Oxford Science Publications, Clarendon Press, Oxford.

[11] Ranta. A. (1988), Constructing possible worlds, Mimeographed, University of Stockholm, to appear in Theoria.

[12] Salvesen. A. (1989), The Church-Rosser Theorem for LF with beta,eta-reductions, Draft.

[13] Smith. J. (1984), An Interpretation of Martin-Löf's Type Theory in a Type-Free Theory of Propositions, *Journal of Symbolic Logic*, Vol. 49, no. 3, 730 - 753.

[14] Statman. R. (1983), λ-definable functionals and β, η-conversion, *Arch. Math. Logic 23*, 21 - 26.