A note on intersection types

Christian Retoré

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

A note on intersection types

Christian Retoré

N° 2431

Décembre 1994

PROGRAMME 2

Rapport de recherche

ISSN 0249-6399

A note on intersection types

Christian Retoré [*]

Programme 2 — Calcul symbolique, programmation et génie logiciel
Projet MEIJE

Rapport de recherche n2431 — Décembre 1994 — 16 pages

Abstract: Following J.-L. Krivine, we call $\mathcal{D}$ the type inference system introduced by M. Coppo and M. Dezani where types are propositional formulae written with conjunction and implication from propositional letters — there is no special constant ω. We show here that the well-known result on $\mathcal{D}$, stating that any term which possesses a type in $\mathcal{D}$ strongly normalises does not need a new reducibility argument, but is a mere consequence of strong normalization for natural deduction restricted to the conjunction and implication. The proof of strong normalization for natural deduction, and therefore our result, as opposed to reducibility arguments, can be carried out within primitive recursive arithmetic. On the other hand, this enlightens the relation between $\&$ and $\hat{\&}$ that G. Pottinger has already wondered about, and can be applied to other situations, like the lambda calculus with multiplicities of G. Boudol.

Key-words: Lambda calculus , intersection types , strong normalization. Logic, proof theory, natural deduction, strong normalization.

(Résumé : tsvp)

* retore@loria.fr

Unité de recherche INRIA Sophia-Antipolis
2004 route des Lucioles, BP 93, 06902 SOPHIA-ANTIPOLIS Cedex (France)
Téléphone : (33) 93 65 77 77 – Télécopie : (33) 93 65 77 65

Une note sur les types avec intersection

Résumé : Suivant la terminologie de J.-L. Krivine, appelons $\mathcal{D}$ le système d'inférence de type introduit par M. Coppo et M. Dezani, dont les types sont des formules propositionnelles écrites avec la conjonction et l'implication intuitionnistes à partir de variables propositionnelles — sans la constante particulière ω. Nous montrons ici que le résultat bien connu sur $\mathcal{D}$, qui affirme que tout terme typable dans $\mathcal{D}$ est fortement normalisable ne nécessite pas un argument de réductibilité, mais est une simple conséquence de la normalisation forte de la déduction naturelle restreinte à l'‘implication et la conjonction. La normalisation forte de la déduction naturelle, et par là même notre résultat, peut être établie dans l'arithmétique primitive récursive, à la différence des arguments de réductibilité. Par ailleurs, on éclaire ainsi la relation entre $\&$ et $\hat{\&}$ déjà envisagée par G. Pottinger, et notre méthode peut s'appliquer à d'autres systèmes comparables tel le lambda calcul avec multiplicités de G. Boudol.

Mots-clé : Lambda calcul, types avec intersection, normalisation forte. Logique, théorie de la démonstration, déduction naturelle, normalisation forte.

A. INTRODUCTION

The type inference system with intersection was introduced by M. Coppo, M. Dezani and P. Sallé [CD78, Sal78, CDC80, CDCV81], to extend Curry's theory of functionality. It involves the usual arrow of Curry's system, together with the conjunction or intersection, and a constant called ω, standing for the empty conjunction. This type inference system, called $\mathcal{D}\Omega$ in [Kri90], allows to characterise solvable λ-terms as λ-terms having a non-trivial type according to $\mathcal{D}\Omega$, and normalising λ-terms as λ-terms having a type without ω in a context without ω according to $\mathcal{D}\Omega$ [CDCV81, Kri90].

The system we are here interested in, is obtained from $\mathcal{D}\Omega$ by leaving out the constant formula ω, introduced in [CDC80, Pot80], called system $\mathcal{D}$ in [Kri90] allows a characterisation of strongly normalising λ-terms as $\mathcal{D}$-typable λ-terms [CDC80, Pot80, Kri90].

We (re)prove here that *"every $\mathcal{D}$-typable λ-term strongly normalises"* , using the strong normalization of natural deduction. It is known that $\mathcal{D}$ is a sub-calculus of natural deduction for intuitionistic logic [Gen34, Jaś34, Pra65] and we use its tree-like presentation of [Pra65] in order to make use of the standard result of its strong normalization [Pra71, Gir87] — thus avoiding a reducibility argument.

This method presents the following advantages:

- ⋆ The reducibility argument usually used for this proof can not be carried out in Primitive Recursive Arithmetic (PRA), while strong normalization for natural deduction can, like in [Gir87] [1]. This proof of [Gir87] reduces strong normalization to weak normalization by PRA means — following the argument that [Gan80] introduced for Gödel's system $\mathcal{T}$. In the ND case the proof of weak normalization is clearly a proof of PRA , and therefore the strong normalization too.

- ⋆ This proof enlightens the relation between the conjunction of $\mathcal{D}$ and intuitionistic conjunction. The former is a restriction of the latter: conjunction is applied only when the *proofs* have the same underlying arrow structure. Because of this restriction according to the term (a part of the already built proof), and not to the involved formulae, $\mathcal{D}$ may not be considered as a logical system like simply typed

[1] The first proof of strong normalization for natural deduction, [Pra71], was using a reducibility argument and could not be carried out in PRA .

λ-calculus, Gödel's system $\mathcal{T}$, Girard's system $\mathcal{F}$... This fact was already underlined by [Hin84][2], and is confirmed by our result: our proof of strong normalization for $\mathcal{D}$-typable terms can be carried out in PRA while $\mathcal{D}$ includes integers and all total recursive functions.

$\star$ Our method is general, hence relevant for other systems, like the λ-calculus with multiplicities [Bou93]. Roughly speaking, it is a refinement of $\mathcal{D}$, where intuitionistic logic for $\to, \wedge$ is replaced by linear logic with weakening for $\multimap, \otimes$.

B. Labelled natural deduction LND

We present $\mathcal{D}$ as a tree-like natural deduction system, namely a sub-calculus of ND [Pra65] because of the two following reasons: the normalization of ND proofs is closer to β-reduction, and strong normalization of the underlying proofs in a "sequent" presentation of natural deduction may not be reckoned as a standard result.

We define a type inference according to $\mathcal{D}$ as a labelled natural deduction LND for short, i.e. a natural deduction where each node is labelled with a λ-term.

We define them in order to have: $x_1 : H_1, ..., x_n : H_n \vdash t : T$ in $\mathcal{D}$ where the free variables of t are exactly the ones in $x_1 : H_1, ..., x_n : H_n$ if and only if there is a LND proof of $t : T$ under the assumptions $x_1 : H_1, ..., x_n : H_n$, i.e. for any free leave of the PLND there exists an index i such that the leave is $x_i : H_i$.

These LND proofs satisfy the following property, needed to define the matching operation:

$\mathcal{L}$: two free leaves labelled by the same variable have the same formula.

Apart from their labels, these proofs are standard natural deductions, although not any natural deduction may be labelled to be a LND proof — see [Hin84].

[2] The note [Hin84] gives a formula true according to intuitionistic logic, which is an empty type according to $\mathcal{D}$. The argument is quite general, and can be adapted to e.g. [Bou93] for the refinement of $\mathcal{D}$ working with multiplicative linear logic.

axiom Given any formula A, and a variable x, $x : A$ is a $\mathtt{LND}$ proof.

matching Assume the $\mathtt{d}^1$ and $\mathtt{d}^2$ are two $\mathtt{LND}$ proofs, whose common free leaves labels are x_i $i \in [1, \mathrm{i}]$, the type of the x_i labelled free leaves being X_i^1 in $\mathtt{d}^1$ and X_i^2 in $\mathtt{d}^2$. Let $\mathtt{d}^1\langle\mathtt{d}^2\rangle$ be obtained from $\mathtt{d}^1$ by writing a projection leading from $x_i : X_i^1 \wedge X_i^2$ to $x_i : X_i^1$ above each free leave $x_i : X^1$ $i \in [1, \mathrm{i}]$ of $\mathtt{d}^1$. Then $\mathtt{d}^1\langle\mathtt{d}^2\rangle$ is a $\mathtt{LND}$ as well. We similarly define $\langle\mathtt{d}^1\rangle\mathtt{d}^2$ obtained from $\mathtt{d}^2$ by adding a projection leading from $x_i : X_i^1 \wedge X_i^2$ — and not $x_i : X_i^2 \wedge X_i^1$ — to $x_i : X_i^2$ above each free $x_i : X^2$.

$\underline{\underline{\mathtt{d}^1\langle\mathtt{d}^2\rangle :}}$

$$\dfrac{x_1 : X_1^1 \wedge X_1^2}{x_1 : X_1^1}\,\pi_1 \qquad \dfrac{x_1 : X_1^1 \wedge X_1^2}{x_1 : X_1^1}\,\pi_1 \qquad \dfrac{x_2 : X_2^1 \wedge X_2^2}{x_2 : X_2^1}\,\pi_1 \qquad \dfrac{x_2 : X_2^1 \wedge X_2^2}{x_2 : X_2^1}\,\pi_1 \qquad \ldots\ldots z_1 : Z_1$$

$$\vdots\ \mathtt{d}^1$$

$$t : T$$

$\underline{\underline{\langle\mathtt{d}^1\rangle\mathtt{d}^2 :}}$

$$\dfrac{x_1 : X_1^1 \wedge X_1^2}{x_1 : X_1^2}\,\pi_2 \qquad \dfrac{x_1 : X_1^1 \wedge X_1^2}{x_1 : X_1^2}\,\pi_2 \qquad \dfrac{x_2 : X_2^1 \wedge X_2^2}{x_2 : X_2^2}\,\pi_2 \qquad \dfrac{x_2 : X_2^1 \wedge X_2^2}{x_2 : X_2^2}\,\pi_2 \qquad \ldots\ldots y_1 : Y_1$$

$$\vdots\ \mathtt{d}^2$$

$$t' : T'$$

abstraction / arrow introduction If $\mathtt{d}$ is a LND , of $t : T$ under the assumptions $x : A, x_1 : H_1, ..., x_n : H_n$, then the proof obtained by an arrow introduction rule, leading to the formula $A \to T$ where *exactly* the A leaves whose label is x are discharged is a LND , whose conclusion is $\lambda x.t : A \to T$.

If the following proof is a LND :

$$\begin{array}{c} x : A \quad x : A \quad y : A \quad z : B \\ \vdots\ \mathtt{d} \\ t : T \end{array}$$

then the following also is a LND :

$$\dfrac{\begin{array}{c} [x : A] \quad [x : A] \quad y : A \quad z : B \\ \vdots\ \mathtt{d} \\ t : T \end{array}}{(\lambda x.t) : A \to T}\ \lambda x$$

application / arrow elimination Assume we have two LND $\mathtt{d}^1$ and $\mathtt{d}^2$, with conclusions $t : U \to V$ and $u : U$; then the proof obtained from $\mathtt{d}^1\langle\mathtt{d}^2\rangle$ and $\langle\mathtt{d}^1\rangle\mathtt{d}^2$ by an arrow elimination rule, leading to V is a LND as well whose conclusion is $(tu) : V$.

If $\mathtt{d}^1$ and $\mathtt{d}^2$ are the following LND :

$$\begin{array}{c|c} \begin{array}{c} x_1 : A_1^1 \quad x_1 : A_1^1 \quad x_2 : A_2^1 \quad x_2 : A_2^1 \quad \ldots \\ \vdots \\ \vdots\ \mathtt{d}^1 \\ \vdots \\ t : U \to V \end{array} & \begin{array}{c} x_1 : A_1^2 \quad x_1 : A_1^2 \quad x_2 : A_2^2 \quad x_2 : A_2^2 \quad \ldots \\ \vdots \\ \vdots\ \mathtt{d}^2 \\ \vdots \\ u : U \end{array} \end{array}$$

then the following also is a LND :

$$\dfrac{\begin{array}{c} \vdots\ \mathtt{d}^1\langle\mathtt{d}^2\rangle \\ f : U \to V \end{array} \qquad \begin{array}{c} \vdots\ \langle\mathtt{d}^1\rangle\mathtt{d}^2 \\ u : U \end{array}}{(tu) : V}\ @$$

product / conjunction introduction Assume we have two LND proofs $\mathtt{d}^1$ and $\mathtt{d}^2$ whose conclusions $u : U^1$ and $u : U^2$ have *the same label*. Then the proof obtained from $\mathtt{d}^1\langle\mathtt{d}^2\rangle$ and $\langle\mathtt{d}^1\rangle\mathtt{d}^2$ by a conjunction introduction rule, is a LND as well whose conclusion is $u : U^1 \wedge U^2$.

If $\mathtt{d}^1$ and $\mathtt{d}^2$ are the following LND ,

$$\begin{array}{c|c} \begin{array}{c} x_1 : A_1^1 \;\; x_1 : A_1^1 \;\; x_2 : A_2^1 \;\; x_2 : A_2^1 \;\; \ldots \\ \vdots\, \mathtt{d}^1 \\ u : U^1 \end{array} & \begin{array}{c} x_1 : A_1^2 \;\; x_1 : A_1^2 \;\; x_2 : A_2^2 \;\; x_2 : A_2^2 \;\; \ldots \\ \vdots\, \mathtt{d}^2 \\ u : U^2 \end{array} \end{array}$$

then the following also is a LND :

$$\dfrac{\begin{array}{c} \vdots\, \mathtt{d}^1\langle\mathtt{d}^2\rangle \\ u : U^1 \end{array} \qquad \begin{array}{c} \vdots\, \langle\mathtt{d}^1\rangle\mathtt{d}^2 \\ u : U^2 \end{array}}{u : U^1 \wedge U^2} <>$$

projection / conjunction elimination Assume $\mathtt{d}^1$ is a LND whose conclusion is $u : U^1 \wedge U^2$. Then the proof obtained from $\mathtt{d}^1$ by a conjunction elimination, leading to a proof of U^i, is a LND as well, whose conclusion is $u : U^i$.

If the following is a LND :

$$\begin{array}{c} \vdots\, \mathtt{d} \\ u : U^1 \wedge U^2 \end{array}$$

so are the following deductions:

$$\begin{array}{c|c} \dfrac{\begin{array}{c} \vdots\, \mathtt{d} \\ u : U^1 \wedge U^2 \end{array}}{u : U^1} \pi^1 & \dfrac{\begin{array}{c} \vdots\, \mathtt{d} \\ u : U^1 \wedge U^2 \end{array}}{u : U^2} \pi^2 \end{array}$$

A LND proof d of $t : T$ corresponds to a type inference in $\mathcal{D}$ of $t : T$:

1. a LND proof of $\vdash\ t : T$ under the assumptions $x_1 : H_1, \ldots, x_n : H_n$ may be viewed as a type inference in $\mathcal{D}$ leading to $x_1 : H_1, \ldots, x_n : H_n \vdash t : T$, and conversely.

2. given a variable x, if the bound variables of t all have distinct names, different from the free variable names, then either:

    (a) x does neither occur in t nor in d

    (b) all d leaves labelled x are free and x is free in t

    (c) all d leaves labelled x are bound and x is bound in t.

But a LND d is also related to its ND structure:

3. the erasement of the labels in d leads to a ND proof $|\mathtt{d}|$.

$\wedge - redex$	$\hat{\rightsquigarrow} - reducts\ to$	$\wedge - reduct$
$\begin{array}{c} \dfrac{\dfrac{\begin{array}{c}\vdots\ \mathtt{d}^1\langle\mathtt{d}^2\rangle \\ u : U^1\end{array} \qquad \begin{array}{c}\vdots\ \langle\mathtt{d}^1\rangle\mathtt{d}^2 \\ u : U^2\end{array}}{u : U^1 \wedge U^2}{<>}}{u : U^1}\pi_1 \\ \vdots\ \mathtt{d} \end{array}$	$\hat{\rightsquigarrow}$	$\begin{array}{c} \vdots\ \mathtt{d}^1\langle\mathtt{d}^2\rangle \\ u : U^1 \\ \vdots\ \mathtt{d} \end{array}$

Fig. 1

4. the formulae of a $\wedge$-redex have the same label (see Fig. 1).

5. the $\hat{\rightsquigarrow}$-reduction, i.e. the reduction of $\wedge$-redexes preserves the typing: given a proof d in LND of $t : T$, under the assumptions $x_1 : H_1, \ldots, x_n : H_n$, whenever $\mathtt{d}\hat{\rightsquigarrow}\mathtt{d}'$, $\mathtt{d}'$ is also a LND proof with the same conclusion and assumptions: the $\hat{\rightsquigarrow}$-reduction does not modify the labels (see Fig. 1).

C. A RESTRICTION ON TYPES: $\mathcal{T}$

A formula F is said to be a prime formula whenever: for any subformula F' of F (possibly $F' = F$), if $F' = A \to B$, then B is not a conjunction — this restriction for $\mathcal{D}$-typings already appeared in e.g. [CDCV81].

A `LND` proof `d` satisfies $\mathcal{T}$ if and only if any formula appearing in `d` is a prime formula.

We can inductively map formulae onto prime formulae, as in [Hin82]:

if T is atomic, then $T^\star = T$

if $T = \bigwedge T_i$, then $T^\star = \bigwedge T_i^\star$

if $T = U \to (\bigwedge T_i)$, then $T^\star = \bigwedge (U^\star \to T_i^\star)$

— although it does not matter, let us say that the bracketting of the $\bigwedge$ in $T^\star$ and T are the same.

We easily obtain by induction on the `LND` structure the two following propositions, the first one being already in [Hin82]:

Proposition 1 *If one has $x_1 : H_1, ..., x_n : H_n \vdash t : T$ in $\mathcal{D}$ then there is a typing $x_1 : H_1^\star, ..., x_n : H_n^\star \vdash t : T^\star$ which is a `LND` proof.*[3]

Proposition 2 *Let `d` be a `LND` proof of $x_1 : H_1, ..., x_n : H_n \vdash t : T$ satisfying $\mathcal{T}$. If $\mathtt{d} \hat{\rightsquigarrow} \mathtt{d}'$ then $\mathtt{d}'$ is a `LND` proof satisfying $\mathcal{T}$ as well.*

[3]The stricto sensu converse does not hold:

Let

$X = A \wedge A'$	$X^\star = X$
$Y = A \to B \to C \wedge A' \to B \to C'$	$Y^\star = Y$
$Z = B \to (C \wedge C')$	$Z^\star = (B \to C) \wedge (B \to C')$

One has $x : X^\star, y : Y^\star \vdash (yx) : Z^\star$ but not $x : X, y : Y \vdash (yx) : Z$.

Nevertheless one has: $x : X, y : Y \vdash \lambda z.((yx)z) : Z$ using a "dummy" $z : B$. This is completely general: a straight forward induction shows that whenever $x_1 : H_1^\star, ..., x_n : H_n^\star \vdash t : T^\star$ then there exists a term t' with $t'\eta t$ such that $x_1 : H_1, ..., x_n : H_n \vdash t' : T$. So if one like [Pot80] explicitly add an (independent) η rule to $\mathcal{D}$ then it holds.

This converse also holds if one considers a subtyping preorder $\leq$ like in [BCDC83, Hin82] or if types are considered up to the equivalence generated by this preoder, like in [Bou93]. This is clear, since such a subtyping preorder includes $(A \to B) \wedge (A \to C) \leq A \to (B \wedge C)$, and works with a rule: if $x_1 : H_1, ..., x_n : H_n \vdash t : T$ and $T \leq U$ then $x_1 : H_1, ..., x_n : H_n \vdash t : U$.

In [BCDC83] it is shown that, however, the addition of the independant η-rule is equivalent to the addition of rules using a subtyping preorder.

D. PARTICULAR LABELLED NATURAL DEDUCTIONS, PLND

We now restrict our attention to particular labelled natural deductions, PLND :

Definition 1 *A* PLND *proof is a* LND *which satisfies:*

$\mathcal{T}$ *: each formula F appearing in the proof is a prime formula*

$\mathcal{N}$ *: the proof contains no $\wedge$ redex, i.e. is $\hat{\rightsquigarrow}$ normal.*

The previous proposition 2 entails the following

Proposition 3 *If* d *is a* LND *satisfying $\mathcal{T}$, and if* $\mathtt{d}\hat{\rightsquigarrow}\mathtt{d}'$ *with* $\mathtt{d}'$ *satisfying $\mathcal{N}$, then* $\mathtt{d}'$ *is a* PLND .

Regarding typability, PLND proofs are as powerfull as $\mathcal{D}$:

Proposition 4 *If a terms admits a typing $t : T$ in $\mathcal{D}$, then it admits a typing $t : T^{\star}$ which is a* PLND *proof.*

Proof: Immediate consequence of the two previous propositions 1 and 3, taking into account that $\hat{\rightsquigarrow}$ is a strongly normalising reduction. $\diamond$

Proposition 5 *Let* d *be a* PLND *, and let $u : A \wedge B$ be a node of* d*. Then the above rule is a product, unless u is a variable.*

Proof: (By contradiction) Take an inner most (higher most) node $u : A \wedge B$ not coming from a product rule, with u not a variable. What rule may precede this rule? It neither may be:

nothing — since u is not a variable,

an abstraction — because of the conjunctive type,

an application — because of the general restriction $\mathcal{T}$ on types

a projection — otherwise the premise would also be a product formula, still labelled by u which is not a variable, and because we choosed a higher most such configuration the above rule would be a product, and there would be an $\wedge$ redex.

Since we excluded the product rule, we have a contradiction. $\diamond$

Proposition 6 *Let* $\mathtt{d}$ *be a* PLND . *Then projections are only applied when the label is a variable.*

Proof: (By contradiction) Assume we have a projection:

$$\dfrac{\begin{array}{c}\vdots\ \mathtt{d}\\ u : U^1 \wedge U^2\end{array}}{u : U^i}\,\pi^i$$

If u were not a variable then, because of the previous proposition, the above rule would be a product, and there would be an $\wedge$-redex. $\diamond$

We thus obtain a kind of normal form for typings: sequence of projections only at the top of the tree, and sequences of products only before the argument part of applications (we do not prove the second part concerning products, since it is obvious, and not needed for our result).

E. Typing in PLND is preserved by β-reduction

We write $\rightsquigarrow$ the usual reduction of natural reduction, consisting in the transitive closure of the union of $\overset{\wedge}{\rightsquigarrow}$ and $\overset{\beta}{\rightsquigarrow}$, the reduction of $\rightarrow$-redexes:

$\rightarrow -redex$	$\overset{\beta}{\rightsquigarrow} -reducts\ to$	$\rightarrow -reduct$
$\dfrac{\dfrac{\begin{array}{c}[A]_x\ \ [A]_x\ \ [A]_y\ \ A\ \ B\\ \vdots\ \mathtt{d}\\ T\end{array}}{A \rightarrow T}\,\lambda x \qquad \begin{array}{c}\vdots\ \mathtt{d}'\\ A\end{array}}{\begin{array}{c}T\\ \vdots\ \mathtt{d}''\end{array}}\,@$	$\overset{\beta}{\rightsquigarrow}$	$\begin{array}{c}\begin{array}{c}\vdots\ \mathtt{d}'\\ A\end{array}\ \ \begin{array}{c}\vdots\ \mathtt{d}'\\ A\end{array}\ \ [A]_y\ \ A\ \ B\\ \vdots\ \mathtt{d}\\ T\\ \vdots\ \mathtt{d}''\end{array}$

Fig. 2

Remark 1 *Remember that $\rightsquigarrow$, although less local than $\hat{\rightsquigarrow}$, satisfies the following: take a subtree* $\mathtt{t}$ *of a* ND *proof* $\mathtt{d}$*, and consider the* $F_{\mathtt{t}}$*-leaves, the bounded leaves of* $\mathtt{t}$ *whose binder lives outside* $\mathtt{t}$*, as free leaves: this makes* $\mathtt{t}$ *an* ND *proof. If* $\mathtt{t} \rightsquigarrow \mathtt{t}'$ *then* $\mathtt{d} \rightsquigarrow \mathtt{d}[\mathtt{t}'/\mathtt{t}]$*; in* $\mathtt{d}[\mathtt{t}'/\mathtt{t}]$ *the images of an* $F_{\mathtt{t}}$ *leave* $x : X$ *under* $\mathtt{t} \rightsquigarrow \mathtt{t}'$ *are bound by the binder of the corresponding* $F_{\mathtt{t}}$ *leave, namely* $\lambda x.(...)$*. This is also clear when thinking of* ND *proofs as simply typed* λ*-terms with products.*

We can now present an improvement of the lemma 1 of [CDC80]:

Proposition 7 *Let* $\mathtt{d}$ *be a proof of* $t : T$ *in* PLND *. If* t' *is a* β*-reduct of* t*,* $t\ \beta\ t'$*, then there exists a* PLND *proof* $\mathtt{d}'$ *of* $t' : T$ *such that* $|\mathtt{d}| \rightsquigarrow |\mathtt{d}'|$ *in* ND *.*

The improvement lies in proving that the type inference of t' is obtained from the one of t by a sequence of reductions of the underlying proof in ND .

Proof: Given any PLND proof $\mathtt{d}$ of $t : T$, and given any redex $((\lambda x.w)u)$ in $t = ((...((\lambda x.w)u)...))$ we show by induction on $\mathtt{d}$ there exists a PLND proof $\mathtt{d}'$ of $t' = ((...(w[u/x])...)) : T$ under the assumption $x_1 : H_1, ..., x_n : H_n$, such that $|\mathtt{d}| \rightsquigarrow |\mathtt{d}'|$ by reviewing all the possilibities for the last rule.

axiom the conclusion of $\mathtt{d}$ is indexed by a variable — since the choosen redex must be a subterm of the final label, this case is excluded.

projection because of proposition 6 the conclusion of $\mathtt{d}$ is indexed by a variable — since the choosen redex must be a subterm of the final label, this case is excluded too.

abstraction if the last rule is an abstraction, leading to $t = \lambda y.s : Y \to S$, the choosen redex lies within its body $s : S$. The induction hypothesis applied to the PLND proof $\mathtt{f}$ leading to $s : S$ in the context $x_1 : H_1, ..., x_n : H_n, y : Y$, provides a proof $\mathtt{f}'$ such that $|\mathtt{f}| \rightsquigarrow |\mathtt{f}'|$ of $s' : S$ in the context $x_1 : H_1, ..., x_n : H_n, y : Y$; applying an abstraction rule to $\mathtt{f}'$ binding the y leaves leads a proof $\mathtt{d}'$ of $t' : T$ in the context $x_1 : H_1, ..., x_n : H_n$. The preliminary remark 1 shows that $|\mathtt{d}| \rightsquigarrow |\mathtt{d}'|$, q.e.d.

product if the last rule is a product leading from $t : R$ and $t : S$ to $t : R \wedge S = T$ in the context $x_1 : H_1, ..., x_n : H_n$, the left (resp. right) subtree is a proof $\mathtt{f}$ (resp. $\mathtt{g}$) of $t : R$ (resp. $t : S$) in the context $x_1 : H_1, ..., x_n : H_n$. We apply induction hypothesis to these two proofs, with the same redex, and we obtain a proof $\mathtt{f}'$ (resp. $\mathtt{g}'$) of $t' : R$ (resp. $t' : S$) in the context $x_1 : H_1, ..., x_n : H_n$, such that $|\mathtt{f}| \rightsquigarrow |\mathtt{f}'|$ (resp. $|\mathtt{g}| \rightsquigarrow |\mathtt{g}'|$). So a product rule may be applied to get a PLND proof $\mathtt{d}'$ of $t : R \wedge S = T$ (there is no need of a matching, it is already done) and it is easily observed, following the preliminary remark 1 that $|\mathtt{d}| \rightsquigarrow |\mathtt{d}'|$.

application if $t = (t_1 t_2)$ is an application,

1. if the redex is in $t_1 : R \to T$, i.e. $t_1 = ((...((\lambda x.w)u)...))$ and $t' = (t'_1 t_2)$ where $t'_1 = ((...(w[u/x])...))$ apply the result to the proof $\mathtt{f}$ of $t_1 : R \to S$, in a smaller context $\Delta \subset x_1 : H_1, ..., x_n : H_n$, with the same choosen redex. This provides a proof $\mathtt{f}'$ of $t'_1 : R \to S$ in the context Δ satisfying $\mathtt{f} \rightsquigarrow \mathtt{f}'$. We can now apply the application rule to get a proof of $t' = (t'_1 t_2) : T$ in the context $x_1 : H_1, ..., x_n : H_n$ — we do not have to match the common free variables at it was already done. Using the remark 1, it is clear that $|\mathtt{d}| \rightsquigarrow |\mathtt{d}'|$.
2. if the redex is in $t_2 : R$, in the right subtree, proceed symmetrically
3. if the redex is this application $t = (\lambda x.w)u$, our requirements on PLND entails that it defines a $\to$-redex in $|\mathtt{d}|$, the underlying natural deduction. Indeed what may be the last rule above $\lambda x.w : U \to V$?

   because of the term $\lambda x.w$ which is not a variable we know there is a rule above it
   because of the type $U \to V$ it may not be a product
   because of the term $\lambda x.w$ it may not be an application
   because of the proposition 6, it may not be a projection

   Therefore it is an abstraction, which together with the following application rule defines a ND redex.
   We reduce it, and replace anywhere in the proof of $w : V$ the label x by u. This certainly defines a LND, which still enjoys $\mathcal{T}$, and which proves $t[u/x] : T$, under the same assumptions. The condition $\mathcal{N}$ may fail, but using proposition 3, the reduction $\hat{\rightsquigarrow}$ leads to a PLND proof of $t[u/x] : T$.

$\diamond$

Theorem *If t is typable in $\mathcal{D}$ then t strongly normalises, and the proof can be carried out in Primitive Recursive Arithmetic.*

Proof: Because of proposition 4, if t is typable in $\mathcal{D}$ by using a proof $\mathtt{f}$ then $\mathtt{f}$ may be turn into a PLND proof $\mathtt{d}$ by PRA means.

Since we want to remain in PRA , we state strong normalization like this: given any λ-term and a typing $\mathtt{d}$ of it, that we can assume to be a PLND proof, the length of its β reduction paths is bounded by some integer $N(t, \mathtt{d})$; we can even say $N(\mathtt{d})$ since $\mathtt{d}$ contains the information t.

Since the ND proof $|\mathtt{d}|$ strongly normalises the length of its reduction paths is bounded by some integer $N(|\mathtt{d}|)$. Because of the previous proposition a sequence of P β-reductions of t give rise to a sequence of at least $P \leadsto$ reduction in ND , and therefore P is bounded by $N(|\mathtt{d}|)$.

The tree manipulations described in this note can obviously be carried out in PRA , in particular the ones leading from the typing in $\mathcal{D}$ to a PLND proof, and, as explained above, the proof of strong normalization for ND also can. Therefore our proof of *"every $\mathcal{D}$-typable λ-term strongly normalises"* also can.

$\diamond$

Acknowledgements: Thanks to Gérard Boudol for very helpfull discussions, and to Mariangiola Dezani for her comments.

References

[BCDC83] Henk Barendregt, Mario Coppo, and Mariangiola Dezani-Ciancaglini. A filter lambda model and the completeness of type-assignment. *Journal of Symbolic Logic*, 48:931–940, 1983.

[Bou93] Gérard Boudol. Lambda calculus with multiplicities. Technical Report 2025, INRIA, 1993.

[CD78] Mario Coppo and Mariangiola Dezani. A new type-asssignment for λ-terms. *Archive für mathematische Logik und Grundlagenforschung*, 19:139–156, 1978.

[CDC80] Mario Coppo and Mariangiola Dezani-Ciancaglini. An extension of the basic functionnality theory for the λ-calculus. *Notre Dame Journal of Formal Logic*, 21:685–683, October 1980.

[CDCV81] Mario Coppo, Mariangiola Dezani-Ciancaglini, and Betti Venneri. Functional characters of solvable terms. *Zeitschrift für matematische Logik und Grundlagen der Mathematik*, 27:44–58, 1981.

[Gan80] Robin O. Gandy. Proofs of strong normalisation. In J.P. Seldin and J.R. Hinley, editors, *To H.B. Curry: essays on combinatory logic, λ calculus and formalism*, pages 457–477, 1980.

[Gen34] Gehrard Gentzen. Untersuchungen über das logische Schließen I. *Mathematische Zeitschrift*, 39:176–210, 1934. Traduction Française de R. Feys et J. Ladrière: Recherches sur la déduction logique, Presses Universitaires de France, Paris, 1955.

[Gir87] Jean-Yves Girard. *Proof-Theory and Logical Complexity — vol. I*. Studies in Proof Theory. Bibliopolis, Napoli, 1987.

[Hin82] J. Roger Hindley. The simple semantics for Coppo-Dezani-Sallé types. In Mariangola Dezani-Ciancaglini and Ugo Montanari, editors, *5^{th} International Symposium on Programming (Torino)*, volume 137 of *Lecture Notes in Computer Science*, pages 212–226. Springer-Verlag, 1982.

[Hin84] J. Roger Hindley. Coppo-Dezani types do not correspond to propositional logic. *Theoretical Computer Science*, 28:235–236, 1984.

[Jaś34] Stanisław Jaśkowski. On the rules of supposition in formal logic. *Studia logica*, 1, 1934.

[Kri90] Jean-Louis Krivine. *Lambda Calcul — Types et Modèles*. Etudes et Recherches en Informatique. Masson, Paris, 1990.

[Pot80] Garrel Pottinger. A type assignment for the strongly normalizable λ-terms. In J.P. Seldin and J.R. Hindley, editors, *To H.B. Curry: Essays on Combinatory Logic,, λ-calculus and formalism*, pages 561–577. Academic Press, 1980.

[Pra65] Dag Prawitz. *Natural Deduction, a Proof-theoretical Study*. Number 3 in Acta universitatis stockholmiensis — Stockholm studies in philosophy. Almqvist and Wiksell, Stockholm, 1965.

[Pra71] Dag Prawitz. Ideas and results in proof theory. In J.E. Fenstadt, editor, *Proceedings of the second scandinavian logic symposium*, Sudies in Logic, pages 235–307, Amsterdam, 1971. North-Holland.

[Sal78] Patrick Sallé. Une extension de la théorie des types en λ-calcul. In G. Ausellio and C. Böhm, editors, *5*$^{\text{th}}$ *International Colloquium on Automata, Languages and Programming (Udine)*, volume 62 of *Lecture Notes in Computer Science*. Springer Verlag, 1978.

Unité de recherche INRIA Lorraine, Technopôle de Nancy-Brabois, Campus scientifique,
615 rue du Jardin Botanique, BP 101, 54600 VILLERS LÈS NANCY
Unité de recherche INRIA Rennes, Irisa, Campus universitaire de Beaulieu, 35042 RENNES Cedex
Unité de recherche INRIA Rhône-Alpes, 46 avenue Félix Viallet, 38031 GRENOBLE Cedex 1
Unité de recherche INRIA Rocquencourt, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex
Unité de recherche INRIA Sophia-Antipolis, 2004 route des Lucioles, BP 93, 06902 SOPHIA-ANTIPOLIS Cedex

Éditeur
INRIA, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex (France)
ISSN 0249-6399