A short proof of the strong normalization of the simply typed $\lambda\mu$-calculus

René David, Karim Nour

A short proof of the strong normalization of the simply typed $\lambda\mu$-calculus

René DAVID and Karim NOUR
LAMA - Equipe de Logique
Université de Chambéry
73376 Le Bourget du Lac
e-mail : {david,nour}@univ-savoie.fr

Abstract. We give an elementary and purely arithmetical proof of the strong normalization of Parigot's simply typed $\lambda\mu$-calculus.

1. Introduction

This paper gives an elementary and purely arithmetical proof of the strong normalization of the cut-elimination procedure for the implicative propositional classical logic, i.e. the propositional calculus with the connectives $\rightarrow$ and $\bot$. As usual, $\bot$ codes the absurdity and the negation is defined by $\neg A = A \rightarrow \bot$.

This proof is based on a proof of the strong normalization of the simply typed λ-calculus due to the first author (see [2]) which, itself, is a simplification of the one given by R. Matthes in [3]. After this paper had been written we were told by P.L. Curien and some others that this kind of technique was already present in van Daalen (see [8]) and J.J. Levy (see [4]).

Since the proofs in the implicative propositional classical logic can be coded by Parigot's $\lambda\mu$-terms and the cut elimination corresponds to the $\lambda\mu$-reduction, the result can be seen as a proof of the strong normalization of the simply typed $\lambda\mu$-calculus. The first proof of the the strong normalization of the $\lambda\mu$-calculus for the types of Girard's system F was done by Parigot in [6] in two different ways : by using reducibility candidates and by a CPS transformation to the λ-calculus.

The technique we present here can also be used to prove the strong normalization of the cut elimination procedure for the classical natural deduction (i.e. where all the connectives, in particular $\vee$, are present and permutative conversions are considered) but more elaborate ideas are necessary. This result was proved (see [1]) by using a CPS transformation. We will give a direct proof in a forthcoming paper.

2. The typed system

The $\lambda\mu$-terms, which extend the λ-terms, are given by the following grammar (where $x, y, ...$ are variables):

$$\mathcal{T} ::= x \mid \lambda x\mathcal{T} \mid (\mathcal{T}\ \mathcal{T}) \mid \mu x\mathcal{T}$$

The new constructor μ corresponds to the classical rule $\bot_c$ given below.

$$\frac{}{\Gamma, x : A \vdash x : A}\, ax \qquad \frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x M : A \to B}\ \to_i$$

$$\frac{\Gamma_1 \vdash M : A \to B \quad \Gamma_2 \vdash N : A}{\Gamma_1, \Gamma_2 \vdash (M\ N) : B}\ \to_e \qquad \frac{\Gamma, x : \neg A \vdash M : \bot}{\Gamma \vdash \mu x M : A}\ \bot_c$$

The cut-elimination procedure corresponds to the reduction rules given below.

A logical cut appears when the introduction of the connective $\to$ is immediately followed by its elimination. The reduction rule is the usual β reduction of the λ-calculus:

$$(\lambda x M\ N) \to M[x := N]$$

A classical cut appears when the classical rule is immediately followed by the elimination rule of $\to$. The reduction rule is :

$$(\mu x M\ N) \to \mu y M[x := \lambda z(y\ (z\ N))]$$

It corresponds to the following transformation on the proofs (written in the natural deduction style):

$$\dfrac{\dfrac{\begin{array}{c}[\neg(A \to B)]\\ \mathcal{D}_1\\ \bot\end{array}}{A \to B} \quad \begin{array}{c}\mathcal{D}_2\\ A\end{array}}{B} \quad \rightsquigarrow \quad \dfrac{\dfrac{\begin{array}{c}[\neg B]\\ \mathcal{D}_1\\ \bot\end{array}}{\neg(A \to B)} \quad \dfrac{\begin{array}{c}\mathcal{D}_2\\ A\end{array}}{[A \to B]}}{B}$$

This coding, though slightly different from the one in [6], is essentially the same and the two systems are obviously equivalent.

- Parigot uses two sets of variables: the λ-variables (for the intuitionistic assumptions) and the μ-variables (for classical assumptions, i.e. the ones that are discharged by the absurdity rule $\bot_c$). Moreover his typing judgements have several conclusions.

- We use only one set of variables and sequents with only one conclusion. Thus, we do not need the new constructor $[\alpha]$ and the corresponding notion of substitution. The drawback is that the reduction introduces some "administrative" redexes. These notations and reductions rules are the ones used in the λ_Δ of Rehof and Sorensen (see [7]).

3. Strong normalization

We first need some notations and lemmas.

3.1. Lemmas for the un-typed calculus

Notation 3..1 *Let M be a $\lambda\mu$-term.*

1. *$M \to M'$ (resp. $M \to^* M'$) means that M reduces to M' by using one step (resp. some steps) of the reduction rules given above.*
2. *$cxty(M)$ is the number of symbols occurring in M.*
3. *M is strongly normalizable (this is denoted by $M \in SN$) if there is no infinite sequence of $\to$ reductions. If $M \in SN$, $\eta(M)$ is the length of the longest reduction of M.*
4. *$\overrightarrow{N}$ (resp. $\overrightarrow{\lambda\mu}$) represents a sequence of $\lambda\mu$-terms (resp. of λ or μ abstractions). If $\overrightarrow{N}$ is the sequence $N_1...N_n$, $(M\ \overrightarrow{N})$ denotes the $\lambda\mu$-term $(M\ N_1...N_n)$.*
5. *In a proof by induction, IH will denote the induction hypothesis.*

Lemma 3..1 *Every $\lambda\mu$-term M can be written as $\overrightarrow{\lambda\mu}(R\ \overrightarrow{O})$ where R is either a redex (called the head-redex of M) or a variable (in this case, M is in head normal form).*

Proof By induction on $cxty(M)$. □

Definition 3..1 *Let M be a $\lambda\mu$-term.*

1. *$hred(M)$ is the term obtained from M by reducing its head-redex, if any.*
2. *$arg(M)$ is the set of terms defined by:*
   - $arg(\overrightarrow{\lambda\mu}(x\ O_1...O_n)) = \{O_1, ..., O_n\}$.
   - $arg(\overrightarrow{\lambda\mu}(\lambda x P\ Q\ O_1...O_n)) = arg(\overrightarrow{\lambda\mu}(\mu x P\ Q\ O_1...O_n)) = \{P, Q, O_1, ..., O_n\}$.

Lemma 3..2 *Let M, N be $\lambda\mu$-terms. Then, $arg(M[x := N]) \subset arg(N) \cup \{N\} \cup \{Q[x := N]\ /\ Q \in arg(M)\}$.*

Proof Immediate. □

Lemma 3..3 *Let M be a $\lambda\mu$-term. Then, $M \in SN$ iff $arg(M) \subset SN$ and $hred(M) \in SN$.*

Proof $\Rightarrow$ is immediate. $\Leftarrow$: If $M = \overrightarrow{\lambda\mu}(x\ \overrightarrow{O})$ the result is trivial.

- If $M = \overrightarrow{\lambda\mu}(R\ \overrightarrow{O})$ where $R = \lambda x P\ Q$: since $arg(M) \subset SN$, an infinite reduction of M must look like: $M \to^* \overrightarrow{\lambda\mu}(\lambda x P_1\ Q_1\ \overrightarrow{O_1}) \to \overrightarrow{\lambda\mu}(P_1[x := Q_1]\ \overrightarrow{O_1}) \to^* ...$. The result immediately follows from the fact that $(P[x := Q]\ \overrightarrow{O}) \to^* (P_1[x := Q_1]\ \overrightarrow{O_1})$.

- If $M = \overrightarrow{\lambda\mu}(R\ \overrightarrow{O})$ where $R = \mu x P\ Q$: the proof is similar. □

Lemma 3..4 *Let $M \in SN$ be $\lambda\mu$-term. Then $(M\, y) \in SN$.*

Proof We prove by induction on $(\eta(M), cxty(M))$ that, if $M \in SN$, then $(M[\sigma]\, y) \in SN$ where σ is a substitution of the form : $[x_1 := \lambda u(x_1\ (u\ y)), ..., x_n := \lambda u(x_n\ (u\ y))]$. It follows immediately from the IH that, if N is a strict sub-term of M, then $N[\sigma] \in SN$ and thus $arg((M[\sigma]\, y)) \subset SN$. By lemma 3..3, it is thus enough to prove that $N = hred((M[\sigma]\, y)) \in SN$. In each case the result follows easily from the IH:

- If $M = (x\ \overrightarrow{O})$ and $\sigma(x) = x$: then $(M[\sigma]\, y) = (x\ \overrightarrow{O[\sigma]}\ y)$.
- If $M = (x\ O_1 \overrightarrow{O})$ and $\sigma(x) = \lambda u(x\ (u\ y)$: then $N = (x\ (O_1[\sigma]\ y)\overrightarrow{O}[\sigma]\ y)$.
- If $M = (\lambda x P\ Q\ \overrightarrow{O})$: then $N = (M'[\sigma]\, y)$ where $M' = hred(M)$ and thus $\eta(M') < \eta(M)$.
- If $M = (\mu x P\ Q\ \overrightarrow{O})$: similar.
- If $M = \lambda x P$: then $N = P[\sigma][x := y]$.
- If $M = \mu x P$: then $N = \mu x P[\sigma']$ where $\sigma' = \sigma \cup [x := \lambda u(x\ (u\ y))]$. □

3.2. Proof of the strong normalization of the typed calculus

The following result is straightforward.

Lemma 3..5 *If $\Gamma \vdash M : A$ and $M \to^* N$ then $\Gamma \vdash N : A$.*

Lemma 3..6 *Let $M \in SN$ be a $\lambda\mu$-term and σ be a substitution. Assume that the substituted variables all have* the same type *and, for all x, $\sigma(x) \in SN$. Then $M[\sigma] \in SN$.*

Proof This is done by induction on $(lgt(\sigma), \eta(M), cxty(M), \eta(\sigma))$ where $lgt(\sigma)$ is the number of connectives in the type of the substituted variables and $\eta(\sigma)$ is the sum of the $\eta(N)$ for the N that are actually substituted, i.e. for example if $\sigma = [x := N]$ and x occurs n times in M, then $\eta(\sigma) = n.\eta(N)$. The cases $M = \lambda x P$, $M = \mu x P$ and $M = (y\ \overrightarrow{P})$ for $y \neq x$ are trivial. Otherwise, by the IH and lemma 3..2, $arg(M[\sigma]) \subset SN$. By lemma 3..3 it is thus enough to show that $hred(M[\sigma]) \in SN$:

- If $M = (\lambda y P\ Q\ \overrightarrow{O})$: $hred(M[\sigma]) = hred(M)[\sigma]$ and the result follows from the IH since $\eta(hred(M)) < \eta(M)$.
- $M = (\mu y P\ Q\ \overrightarrow{O})$: similar.
- $M = (x\ P\ \overrightarrow{O})$: by our definition of $\eta(\sigma)$, we may assume, without loss of generality, that x occurs only once in M. Let $N = \sigma(x)$.
  - If N is not in head normal form, $hred(M[\sigma]) = M[\sigma']$ where $\sigma'(y) = \sigma(y)$ for $y \neq x$ and $\sigma'(x) = hred(N)$. The result follows from the IH since $\eta(\sigma') < \eta(\sigma)$.
  - If $N = (y\ \overrightarrow{N_1})$, the result is trivial.

– If $N = \lambda y N_1$ then $hred(M[\sigma]) = (N_1[y := P[\sigma]]\ \overrightarrow{O[\sigma]})$. By the IH, since $lgt(P[\sigma]) < lgt(\sigma)$, $N_1[y := P[\sigma]] \in SN$ and thus, by the IH, $hred(M[\sigma]) = (z\ \overrightarrow{O[\sigma]})\ [z := N_1[y := P[\sigma]]] \in SN$ since $lgt(N_1) < lgt(\sigma)$.

– If $N = \mu y N_1$ then $M[\sigma] = (\mu y N_1\, P[\sigma]\, \overrightarrow{O[\sigma]})$. Let $M_1 = (\mu y N_1\, z)$ where z is a fresh variable. By lemma 3..4, $M_1 \in SN$. Since $lgt(P[\sigma]) < lgt(\sigma)$, by the IH, $(\mu y N_1\, P[\sigma]) = M_1[z := P[\sigma]] \in SN$. But $M[\sigma] = M_2[u := (\mu y N_1\, P[\sigma])]$ where $M_2 = (u\, \overrightarrow{O[\sigma]})$ and u is a fresh variable. Since $lgt((\mu y N_1\, P[\sigma])) < lgt(\sigma)$, the result follows from the the IH. □

Theorem 3..1 *Every typed $\lambda\mu$-term is strongly normalizable.*

Proof By induction on $cxty(M)$. The cases $M = x$, $M = \lambda x\ N$ or $M = \mu x\ N$ are trivial. If $M = (N_1\ N_2)$ the result follows from lemma 3..6 and the IH since $M = (x\ N_2)[x := N_1]$ where x is a fresh variable. □

Remarks

1. In the proof of theorem 3..1, the case $M = (N_1\ N_2)$ can also be solved, by writing $M = (N_1\ x)[x := N_2]$ where x is a fresh variable and using lemma 3..3.

2. In the proof of lemma 3..4, the case $M = (x\ P\ \overrightarrow{O})$ and $N = \lambda y N_1$ can be solved exactly as the case $N = \mu y N_1$ by using $M_1 = (\lambda y N_1\, z)$ where z is a fresh variable.

4. References

[1] P. de Groote. *Strong normalization of classical natural deduction with disjunction.* TLCA'01 in Lecture Notes in Computer Science (2044), pp. 182-196. Springer Verlag, 2001.

[2] R. David. *Normalization without reducibility.* Annals of Pure and Applied Logic (107), p. 121-130, 2001.

[3] F. Joachimski and R. Matthes. *Short proofs of normalization for the simply-typed lambda-calculus, permutative conversions and Gödel's T* in Arch. Math. Logic 42, p 59-87 (2003).

[4] J.J. Levy. *Rductions correctes et optimales dans le lambda-calcul.* PhD thesis Paris 7, 1978.

[5] M. Parigot *$\lambda\mu$-calculus: An algorithm interpretation of classical natural deduction.* Lecture Notes in Artificial Intelligence (624), pp. 190-201. Springer Verlag 1992.

[6] M. Parigot. *Proofs of strong normalization for second order classical natural deduction.* Journal of Symbolic Logic, 62 (4), pp. 1461-1479, 1997.

[7] N.J. Rehof and M.H. Sorensen. *The λ_Δ-calculus.* TACS'94 in Lecture Notes in Computer Science (789), pp. 516-542. Springer Verlag, 1994.

[8] D. van Daalen *The language theory of Automath.* PhD Thesis. Eindhoven 1977.