Bulletin of the **S**ection of **L**ogic
Volume 45/1 (2016), pp. 1–15

http://dx.doi.org/10.18778/0138-0680.45.1.01

Fernando Ferreira and Gilda Ferreira

AN ELEMENTARY PROOF OF STRONG NORMALIZATION FOR ATOMIC F^*

Abstract

We give an elementary proof (in the sense that it is formalizable in Peano arithmetic) of the strong normalization of the atomic polymorphic calculus F_{at} (a predicative restriction of Girard's system F).

Keywords: Predicative polymorphism, strong normalization, elementary proofs, lambda-calculus.

1. Introduction

It is well-known that the (impredicative) polymorphic system F of Jean-Yves Girard enjoys the property of strong normalization. A particularly perspicuous description of Girard's system and attendant proof of (strong) normalization can be found in [7]. The normalization proof is quite involved, using the notion of "reducibility candidate." Moreover, it is not formalizable in full second-order arithmetic (a very strong system). This is not a defect of Girard's proof: No such proof can be formalized in full second-order arithmetic since that would provide a consistency proof of this system within itself, contradicting Gödel's second incompleteness theorem. In 2006, the first author introduced the system F_{at} of atomic polymorphism [4][1]. Basically, F_{at} is the system obtained from Girard's F by restricting

*2010 *Mathematics Subject Classification.* 03F07, 03B20, 03B40

[1]Paper [4] uses a different terminology. It denotes by atomic $PSOL^i$ what we now call system F_{at}.

the range of type variables to *atomic* types. Although this is a severe restriction, F_{at} is able to embed the full intuitionistic propositional calculus IPC (i.e., intuitionistic logic with implication, conjunction, disjunction and falsum). Moreover, F_{at} is a system with good proof-theoretic properties and proved to be a natural (and useful) framework for studying intuitionistic propositional logic. Elegant alternative proofs of properties of full IPC, such as strong normalization or the disjunction property were obtained via its embedding into F_{at} (see [5, 6]). Note that, contrarily to IPC, F_{at} has no "bad" connectives (see Girard's criticism on some natural deduction IPC rules in [7], p. 74) and no commuting conversions. Moreover, due to the restriction on the range of type variables, system F_{at} is predicative (as opposed to F), has a natural notion of subformula (subtype) and enjoys the subformula property.

Since Girard's system F is strongly normalizing, F_{at} inherits this property. But strong normalization for the atomic polymorphic calculus can be proved in a much simpler way, avoiding the intricate resources needed for system F. The authors proved in [5], in a *predicative* manner, that F_{at} enjoys the property of strong normalization for $\beta\eta$-conversions. Our proof uses William Tait's technique of *reducibility*, as introduced in [10]. Of course, Girard's reducibility candidates are not used in the proof (they are only needed to deal with *impredicativity*). That notwithstanding, the proof is not elementary in the sense of not being formalizable in Peano arithmetic. This is a feature of Tait's technique. In [5] we can read [referring to strong normalization in F_{at}]: "*The concrete proof that we presented... is not formalizable in Peano arithmetic (let alone in primitive recursive arithmetic). This is incidental, we think. It would be nice to investigate whether, for instance, the proof technique used by Felix Joachimski and Ralph Matthes in [8] also applies to our system* F_{at} *of atomic polymorphism*". We show in this paper that there is, in fact, a proof of the strong normalization of F_{at} formalizable in Peano arithmetic, and the strategy of our proof relies, precisely, on Joachimski and Matthes' technique: It is possible to extend their argument in [8] to the second order cases typical of atomic polymorphism.

The paper is organized as follows. In the next section, we describe the system F_{at} and introduce the main concepts of this paper. System F_{at} is constituted by typed terms and these are generated in the usual way. In Section 3, we describe an *unusual* way of enumerating the terms of F_{at}. It is not trivial to show that this alternative construction gives exactly the terms of F_{at}. The advantage of the enumeration is that it is suitable for showing,

by induction on the new build up, that all terms are strongly normalizable for $\beta\eta$-conversions. This is done in Section 4. In the final section, we make some comments about elementary *versus* finitistic proofs and raise some questions. It seems to us that proofs of strong normalization using the technique followed here (via the new enumeration of terms), although formalizable in Peano arithmetic, are not finitistic in the sense of not being formalizable in primitive recursive arithmetic.

2. A predicative variant of Girard's system F

The system F_{at}, like Girard's system F, has only two generators for types (formulas): implication and second-order universal quantification. *Types* are constructed from atomic types (propositional constants $P, Q, R, \ldots$ and type variables $X, Y, Z, \ldots$) by means of two type-forming operations, $\to$ and $\forall$, in the following way:

(i) Atomic types are types.
(ii) If A and B are types then $A \to B$ is a type.
(iii) If A is a type and X is a type variable then $\forall X.A$ is a type.

By regarding types as formulas, we have the usual definitions of *free* and *bound* (type) variables in a type. As usual, we can freely rename the bound variables in a type. Given a type A, a type variable X and an *atomic* type C, we write $A[C/X]$ for the type obtained from A by substituting the free occurrences of X in A by C (without loss of generality, if C is itself a variable, we may assume that it is free for X in A). The *terms* of F_{at} are generated by the following clauses:

(i) For each type A there are countably infinite many *assumption variables* of type A, x^A, y^A, z^A, etc. Assumption variables are terms.
(ii) If $r^{A\to B}$ and s^A are terms of types $A \to B$ and A, respectively, then $(r^{A\to B}s^A)^B$ is a term of type B.
(iii) If r^B is a term of type B and x^A is an assumption variable of type A, then $(\lambda x^A.r^B)^{A\to B}$ is a term of type $A \to B$.
(iv) If $r^{\forall X.A}$ is a term of type $\forall X.A$ and C is an *atomic* type, then $(r^{\forall X.A}C)^{A[C/X]}$ is a term of type $A[C/X]$.
(v) If r^A is a term of type A and the type variable X is not free in the type of any free assumption variable of r^A, then $(\Lambda X.r^A)^{\forall X.A}$ is a term of type $\forall X.A$.

The term-clause which distinguishes F_{at} from Girard's system F is clause (iv) above. In Girard's system, C can be *any* type. This difference explains the impredicativity of system F vis-à-vis the predicativity of F_{at}. Since in F we can instantiate the universal types $\forall X.A$ by any type D (however complex), obtaining $A[D/X]$, Girard's system has no sensible notion of subformula. In contrast, there is a natural notion of subformula in system F_{at}: the immediate subformulas of $\forall X.A$ are the formulas of the form $A[C/X]$, where C is an atomic type (free for X in A). By regarding types as formulas (Curry-Howard isomorphism) we write "A is a subtype of B" meaning "A is a subformula of B". More precisely:

Definition 1. *The subtypes of a type A are defined by:*

(i) *A is a subtype of A.*
(ii) *If $B \to C$ is a subtype of A then B and C are both subtypes of A.*
(iii) *If $\forall X.B$ is a subtype of A then $B[C/X]$ is a subtype of A, for all atomic type C free for X in B.*

In order to denote that the term r is of type A, it is usual to write r^A or $r : A$ (note that our formalism has rigid typing, i.e. every term carries a fixed type). When the type is clear from the context, or need not be specified, we simply write r. We presuppose as known the notion of the set of free (assumption and type) variables of a term r, denoted by $\mathsf{FV}(r)$. We consider all expressions modulo renaming of bound variables. We also presuppose as known the notion of substitution of a free assumption variable x^A in a term r by a term s^A, denoted by $r[s/x]$, and the notion of substitution of a free type variable X in a term r^B by an atomic type C, denoted by $r[C/X]$ (of type $B[C/X]$). We always assume that there are no clashes of variables in the substitutions (if needed, bound variables are renamed). Details can be found in [5].

In analogy with system F, we have two β-conversions: one for implication, the *arrow β-conversion*, and the other for second-order universal quantification, the *universal β-conversion*. They are, respectively,

$$\begin{array}{lcl} (\lambda x.r)s & \rightsquigarrow & r[s/x] \\ (\Lambda X.r)C & \rightsquigarrow & r[C/X], \end{array}$$

where the left-hand side of a conversion is called its *redex* and the right-hand side its *contractum*. Note that 'C' above stands for an atomic type. We also use in the sequel the so-called η-conversions:

$$\begin{array}{rcll} \lambda x.(rx) & \rightsquigarrow & r, & \text{with } x \notin \mathsf{FV}(r) \\ \Lambda X.(rX) & \rightsquigarrow & r, & \text{with } X \notin \mathsf{FV}(r). \end{array}$$

The first one is the *arrow η-conversion*, the second is the *universal η-conversion*. As before, terms on the left-hand side are called *redexes* and on the right-hand side are called *contracta*.

DEFINITION 2. *A term r reduces to a term s in one step, and we write, $r \succ_1 s$, if s is obtained from r by replacing a redex by its contractum. We say that a term r reduces to a term s (and we write $r \succeq s$) if there is a sequence of $\beta\eta$-conversions from r to s, i.e., a sequence $r \equiv u_0, u_1, \ldots, u_n \equiv s$, such that for $i = 0, 1, \ldots, n-1$, $u_i \succ_1 u_{i+1}$. A term is normal if it has no redexes and so we can no longer apply a conversion. A term r is strongly normalizable if all the reduction sequences starting with r have finite length.*

The objective of this paper is to give an elementary proof of the fact that all terms of F_{at} are strongly normalizable for $\beta\eta$-conversions. Note, however, that the statement of strong normalization of a given term of F_{at} (we assume an arithmetization of the syntax of F_{at}) is *prima facie* second-order and, therefore, not expressible in the first-order language of PA. The statement says that there is no *infinite* reduction sequence starting at the given term (the negated existential part is second-order). For the expert, it is a strict Π^1_1-statement. It is well-known that strict Π^1_1-statements are equivalent to Σ_1-statements (this follows from the so-called König's lemma). Therefore, the statement that all terms of F_{at} are strongly normalizable is equivalent to a Π_2-sentence. In fact, and as it is well-known, it can be put in the following form: for all terms t of F_{at} there is a *finite* reduction tree for t. The reduction tree for a term t is constituted by the collection of finite reduction sequences starting from t, partially ordered under the initial part relation (see [12], pages 12-13, for more information).

In the sequel we prove theorems in the usual mathematical style, without caring for their formalizations.

3. On enumerating the terms of F_{at}

In this section, we present a particular enumeration of the terms of F_{at}. This enumeration is based on work of Joachimsky and Matthes to prove some strong normalization results (and was first presented in [15] as a positive inductive definition of the strongly normalizable terms of the untyped

λ-calculus). The set **SN** is constituted by terms of F_{at} and its elements are obtained according to certain rules. We find that the terminology '**SN**' is not felicitous because it is not obvious from the rules that the terms so generated are strongly normalizable (this is indeed so, but we only prove it in the next section). However, this is the standard terminology and we did not want to change it.

Before describing the rules to obtain the terms in **SN**, we must establish some notation. The notation $\bar{q}$ will be used often in the sequel. It stands for a sequence $q_1, \ldots, q_n$ whose entries are terms or *atomic types*. When we write $\bar{q} \in \mathbf{SN}$ we mean that the *term* entries of the sequence $\bar{q}$ are in **SN**. If t is a further term and the types match, $t\bar{q}$ is the term $(\ldots(tq_1)\ldots q_n)$. We can now give the rules for obtaining the terms of **SN** (it is, of course, understood that types always match and the resulting terms are always well-formed):

$$\frac{\bar{q} \in \mathbf{SN}}{x\bar{q} \in \mathbf{SN}}\ (var) \qquad \frac{r \in \mathbf{SN}}{\lambda x.r \in \mathbf{SN}}\ (\lambda) \qquad \frac{r[s/x]\bar{q} \in \mathbf{SN} \qquad s \in \mathbf{SN}}{(\lambda x.r)s\bar{q} \in \mathbf{SN}}\ (\beta_{\to})$$

$$\frac{r \in \mathbf{SN}}{\Lambda X.r \in \mathbf{SN}}\ (\Lambda) \qquad \frac{r[C/X]\bar{q} \in \mathbf{SN}}{(\Lambda X.r)C\bar{q} \in \mathbf{SN}}\ (\beta_{\forall})$$

The letter x above is an assumption variable, X is a type variable, r and s are terms, and C is an atomic type. The rule (var) is a multiple premise rule. Note that the sequence $\bar{q}$ can be the empty sequence. Therefore, assumption variables are in **SN**. Here is another example. The sequence $\bar{q}$ could stand for the two-entry sequence C, t where C is an atomic type and t is a term of a certain given type A. If $x : \forall X(A \to X)$, then we can conclude that $(xC)t \in \mathbf{SN}$ from the single premise $t \in \mathbf{SN}$ (and the indication of the sequence C, t). The rules (λ), (Λ) and $(\beta_{\forall})$ are single premise rules and, of course, $(\beta_{\to})$ is a two premise rule.

The following simple combinatorial properties concerning substitution will be used in the proof of the two lemmas below.

FACT 1. *Let r, s and t be terms of F_{at}, A, C and D types, with C and D atomic, x and y distinct assumption variables and X and Y distinct type variables. Suppose also that $x \notin \mathsf{FV}(t)$ and D is not the variable X. Then, whenever the types match,*

a) $(r[s/x])[t/y]$ *is the term* $(r[t/y])[s[t/y]/x]$.
b) $(r[C/X])[D/Y]$ *is the term* $(r[D/Y])[C[D/Y]/X]$.
c) $(r[s/x^A])[C/X]$ *is the term* $(r[C/X])[s[C/X]/x^{A[C/X]}]$.

LEMMA 1. *Take* $r \in \mathbf{SN}$, *D an atomic type and Y a type variable. Then, whenever the types match,*

1. $rD \in \mathbf{SN}$
2. $r[D/Y] \in \mathbf{SN}$.

PROOF: The proof is by simultaneous induction on the build-up of r according to the rules of **SN**. Without loss of generality, we may suppose that free and bound variables are distinct in the terms below.

(var) Suppose that the result is valid for the terms in $\bar{q} \in \mathbf{SN}$. We want to see that it is valid for $x^A\bar{q}$. For (1), note that by (var), $x^A\bar{q}D \in \mathbf{SN}$ (just add one more atomic type to the tuple). For (2), by induction hypothesis, we have $q_k[D/Y] \in \mathbf{SN}$ for each term q_k in $\bar{q}$. So, by (var), $x^{A[D/Y]}(\bar{q}[D/Y]) \in \mathbf{SN}$. (Of course, $\bar{q}[D/Y]$ is the sequence whose entries are obtained from the entries of $\bar{q}$ by performing the indicated substitution.) Note that the term $x^{A[D/Y]}(\bar{q}[D/Y])$ is $(x^A\bar{q})[D/Y]$.

(λ) Let us analyze the case in which $\lambda x^A.r \in \mathbf{SN}$ is generated from $r \in \mathbf{SN}$. The case (1) never occurs due to type restrictions. Concerning (2), by induction hypothesis, we have that $r[D/Y] \in \mathbf{SN}$. Consequently, by (λ), $\lambda x^{A[D/Y]}.(r[D/Y]) \in \mathbf{SN}$. This term is $(\lambda x^A.r)[D/Y]$.

$(\beta_\rightarrow)$ By induction hypothesis, $r[s/x^A]\bar{q}D \in \mathbf{SN}$. By $(\beta_\rightarrow)$, we get (1), i.e. $(\lambda x^A.r)s\bar{q}D \in \mathbf{SN}$. To see (2), we use the induction hypothesis to conclude that the terms $(r[s/x^A]\bar{q})[D/Y]$ and $s[D/Y]$ are in **SN**. The first term is $(r[D/Y][s[D/Y]/x^{A[D/Y]}])(\bar{q}[D/Y])$. By $\beta_\rightarrow$, we conclude that $(\lambda x^{A[D/Y]}.r[D/Y])s[D/Y](\bar{q}[D/Y]) \in \mathbf{SN}$, i.e. $((\lambda x^A.r)s\bar{q})[D/Y] \in \mathbf{SN}$.

(Λ) This is the case in which $\Lambda X.r \in \mathbf{SN}$ is obtained from $r \in \mathbf{SN}$. To see that $(\Lambda X.r)D \in \mathbf{SN}$ it is enough (by $(\beta_\forall)$, with empty sequence $\bar{q}$) that $r[D/X] \in \mathbf{SN}$. But this is our induction hypothesis with type variable X. For (2), by induction hypothesis, $r[D/Y] \in \mathbf{SN}$ and so, by (Λ), we conclude that $\Lambda X.(r[D/Y]) \in \mathbf{SN}$. Note that this term is $(\Lambda X.r)[D/Y]$.

$(\beta_\forall)$ Suppose that (1) and (2) are valid for $r[C/X]\bar{q} \in \mathbf{SN}$. We want to prove that they are also valid for $(\Lambda X.r)C\bar{q} \in \mathbf{SN}$. By induction hypothesis, $r[C/X]\bar{q}D \in \mathbf{SN}$. Hence, by $(\beta_\forall)$, $(\Lambda X.r)C\bar{q}D \in \mathbf{SN}$. For (2), by induction hypothesis, $(r[C/X]\bar{q})[D/Y] \in$

SN. I.e., $r[D/Y][C[D/Y]/X](\bar{q}[D/Y]) \in \mathbf{SN}$. By $(\beta_\forall)$, we get $(\Lambda X.r[D/Y])C[D/Y](\bar{q}[D/Y]) \in \mathbf{SN}$. Note that this term is, precisely, $((\Lambda X.r)C\bar{q})[D/Y]$. □

LEMMA 2. *Take $r \in \mathbf{SN}$, $t^\rho \in \mathbf{SN}$ and y an assumption variable. Then, whenever the types match,*

1. $rt \in \mathbf{SN}$
2. $r[t/y] \in \mathbf{SN}$.

PROOF: The proof is by simultaneous induction, with a main induction on the type ρ and side induction on the build-up of r according to the rules of **SN**. Without loss of generality, we may suppose that free and bound variables are distinct in the terms below.

(var) Suppose that the result is valid for the terms in $\bar{q} \in \mathbf{SN}$. We want to see that it is valid for $x\bar{q}$. Case (1) is very easy, similar to the corresponding one in the previous lemma. The second case is more subtle. By induction hypothesis, $q_k[t/y] \in \mathbf{SN}$ for each q_k a term in the sequence $\bar{q}$. Applying (var), we conclude that $x(\bar{q}[t/y]) \in \mathbf{SN}$. If x and y are different variables, note that $x(\bar{q}[t/y])$ is the term $(x\bar{q})[t/y]$. Otherwise, we need to prove that $t(\bar{q}[t/y]) \in \mathbf{SN}$. But this follows by multiple applications of (1) of Lemma 1 and of the *main induction hypothesis*. Do observe that the terms $q_k[t/y]$ have a type which is a subtype of ρ (types are supposed to match in term applications).

(λ) This is the case in which $\lambda x.r$ is generated from $r \in \mathbf{SN}$. For (1), we want to prove that $(\lambda x.r)t \in \mathbf{SN}$. By $(\beta_\to)$ (with empty sequence $\bar{q}$), it suffices to prove that $r[t/x] \in \mathbf{SN}$. But this is exactly the side induction hypothesis. For (2), notice that the side induction hypothesis yields $r[t/y] \in \mathbf{SN}$ and so, by (λ), $\lambda x.(r[t/y]) \in \mathbf{SN}$. This term is $(\lambda x.r)[t/y]$, as wanted.

$(\beta_\to)$ Let us consider the case in which $(\lambda x.r)s\bar{q}$ is obtained from $r[s/x]\bar{q} \in \mathbf{SN}$ and $s \in \mathbf{SN}$. For (1), note that by the side induction hypothesis we have $r[s/x]\bar{q}t \in \mathbf{SN}$ and so, by $(\beta_\to)$, we get $(\lambda x.r)s\bar{q}t \in \mathbf{SN}$. For (2), by side induction hypothesis, we have $(r[s/x]\bar{q})[t/y] \in \mathbf{SN}$, i.e., $r[t/y][s[t/y]/x](\bar{q}[t/y]) \in \mathbf{SN}$. Since we also have $s[t/y] \in \mathbf{SN}$ by side induction hypothesis, we can conclude that $(\lambda x.r[t/y])s[t/y](\bar{q}[t/y]) \in \mathbf{SN}$ by an application of $(\beta_\to)$. Note that this latter term is $((\lambda x.r)s\bar{q})[t/y]$, as wanted.

(Λ) When $\Lambda X.r \in \mathbf{SN}$ is obtained from $r \in \mathbf{SN}$, the term $(\Lambda X.r)t$ can never be formed because of incompatibility of types. For (2), we want to prove that $(\Lambda X.r)[t/y] \in \mathbf{SN}$. This latter term is $\Lambda X.r[t/y]$. The desired conclusion now follows immediately from the side induction hypothesis $r[t/y] \in \mathbf{SN}$ via an application of (Λ).

($\beta_\forall$) The last case is when $(\Lambda X.r)C\bar{q}$ is obtained from $r[C/X]\bar{q} \in \mathbf{SN}$. By side induction hypothesis, we have $r[C/X]\bar{q}t \in \mathbf{SN}$. Applying ($\beta_\forall$), we obtain (1), i.e. $(\Lambda X.r)C\bar{q}t \in \mathbf{SN}$. For (2), consider the side induction hypothesis $(r[C/X]\bar{q})[t/y] \in \mathbf{SN}$. We claim that this term is $r[t/y][C/X](\bar{q}[t/y])$. This is clear if y does not occur free in r. If y occurs free in r, use (c) of Fact 1 (note that by the rule of formation of second-order abstraction the variable X cannot occur free in the type of y). In short, $r[t/y][C/X](\bar{q}[t/y]) \in \mathbf{SN}$. By rule ($\beta_\forall$), we get $(\Lambda X.r[t/y])C(\bar{q}[t/y]) \in \mathbf{SN}$. The latter term is $((\Lambda X.r)C\bar{q})[t/y]$, as wanted. □

THEOREM 1. *All the terms of* F_{at} *are in* **SN**.

PROOF: This is a consequence of the fact that assumption variables are in **SN**, of the rules (λ) and (Λ), and of the two previous lemmas. □

4. Terms have finite reduction trees

Let us denote by $t \in \mathbf{FR}$ the statement that the term t has a finite reduction tree. Given a finite tree, we can associate to each node of the tree its height (relative to the tree). The height of a leaf is just zero. The height of an inner node is just the maximum height of its (finitely many) sons plus one. Of course, in a reduction tree, to each node it is associated a term. In the sequel, when we speak of terms in a reduction tree we mean any node in the tree with that term associated.

Most of the following results are well-known:

LEMMA 3. *Let r be a term of* F_{at}. *Suppose that for all terms s such that $r \succ_1 s$, we have $s \in \mathbf{FR}$. Then $r \in \mathbf{FR}$.*

LEMMA 4. *In the following, we presuppose matching types (when appropriate):*

1. *If $r \succ_1 s$, then $q[r/x] \succeq q[s/x]$. Hence, if $r \succeq s$, then $q[r/x] \succeq q[s/x]$.*

2. *If* $q \succ_1 r$, *then* $q[s/x] \succ_1 r[s/x]$. *Hence, if* $q \succeq r$, *then* $q[s/x] \succeq r[s/x]$.
3. *If* $q \succ_1 r$, *then* $q[C/X] \succ_1 r[C/X]$. *Hence, if* $q \succeq r$, *then* $q[C/X] \succeq r[C/X]$.

PROPOSITION 1. *Let* r, s *and* q *be terms of* F_{at}, C *an atomic type,* x *an assumption variable and* X *a type variable. Then, as long as types match appropriately,*

1. *If* $rq \in \mathbf{FR}$ *then* $r \in \mathbf{FR}$ *and* $q \in \mathbf{FR}$.
2. *If* $rC \in \mathbf{FR}$ *then* $r \in \mathbf{FR}$.
3. *If* $r[s/x] \in \mathbf{FR}$ *then* $r \in \mathbf{FR}$.
4. *If* $r[C/X] \in \mathbf{FR}$ *then* $r \in \mathbf{FR}$.

Given the results of the previous section, in order to show that the calculus F_{at} enjoys the property of strong normalization, it is enough to prove the following:

THEOREM 2. *If* $t \in \mathbf{SN}$, *then* $t \in \mathbf{FR}$.

PROOF: The proof is by induction on the build-up of t according to the rules of **SN**. There are five cases to consider.

For (var), assume that $\bar{q} \in \mathbf{FR}$ in the sense that every term in the tuple (if any) has a finite reduction tree. We want to prove that $x\bar{q} \in \mathbf{FR}$. That is immediate since each reduction on $x\bar{q}$ has to take place in the terms in $\bar{q}$ and these are in finite number and have finite reduction trees.

For (λ), let us fix $r_0 \in \mathbf{FR}$. We want to show that $\lambda x.r_0 \in \mathbf{FR}$. We prove that $\lambda x.r \in \mathbf{FR}$ for all terms r in the reduction tree of r_0 by induction on the height of r. Let us consider the one-step reductions of $\lambda x.r$. It can be a reduction to $\lambda x.r'$, where $r \succ_1 r'$. In this case, $\lambda x.r' \in \mathbf{FR}$ by induction hypothesis. The remaining possibility is when r is of the form vx and the term $\lambda x.r$ η-converts to v. Since $vx \in \mathbf{FR}$, by Proposition 1 (1), we conclude that $v \in \mathbf{FR}$. Note that the proof is supported by Lemma 3.

For $(\beta_{\to})$, we assume that $r[s/x]\bar{q} \in \mathbf{FR}$ and $s \in \mathbf{FR}$ in order to show that $(\lambda x.r)s\bar{q} \in \mathbf{FR}$. The proof is by induction on the sum of the heights of r, s and $\bar{q}$. (Note that by hypothesis and by Proposition 1, the terms r, s and the terms in $\bar{q}$ have finite reduction trees.) The possible reductions in one step from $(\lambda x.r)s\bar{q}$ are: i) $(\lambda x.r')s\bar{q}$ with $r \succ_1 r'$; ii) $(\lambda x.r)s'\bar{q}$ with

$s \succ_1 s'$; iii) $(\lambda x.r)s\bar{q}'$ with a term q_i in $\bar{q}$ reducing in one step to a term q_i'; iv) $r[s/x]\bar{q}$; v) $us\bar{q}$ when $r :\equiv ux$. By induction hypothesis, the terms in the first three cases are in **FR**. Note that in all the three cases the induction hypothesis applies (Lemma 4 and/or the closure of **FR** under reduction ensure that: $r'[s/x]\bar{q} \in \mathbf{FR}$, $r[s'/x]\bar{q} \in \mathbf{FR}$, $s' \in \mathbf{FR}$ and $r[s/x]\bar{q}' \in \mathbf{FR}$). The fourth case follows by hypothesis ($r[s/x]\bar{q} \in \mathbf{FR}$) and the last case reduces to the previous one. Thus, by Lemma 3, $(\lambda x.r)s\bar{q} \in \mathbf{FR}$.

The case (Λ) is like the case (λ) above.

Finally, the case ($\beta_\forall$) is very similar to ($\beta_\to$). Let us assume that $r[C/X]\bar{q} \in \mathbf{FR}$ having in view to prove that $(\Lambda X.r)C\bar{q} \in \mathbf{FR}$. The proof is by induction on the sum of the heights of r and $\bar{q}$, possible because, by Proposition 1, r and the terms in $\bar{q}$ have finite reduction trees. The possible reductions in one step from $(\Lambda X.r)C\bar{q}$ are: i) $(\Lambda X.r')C\bar{q}$ with $r \succ_1 r'$; ii) $(\Lambda X.r)C\bar{q}'$ with a term q_i in $\bar{q}$ reducing in one step to a term q_i'; iii) $r[C/X]\bar{q}$; iv) $uC\bar{q}$ when $r :\equiv uX$. The first two cases follow by induction hypothesis, the third case follows by hypothesis and the fourth case reduces to the third. □

As a consequence of Theorems 1 and 2, we have the following result:

Main Theorem. *All the terms of* F_{at} *are strongly normalizable with respect to* $\beta\eta$*-conversions.*

5. Final comments

It is clear that the above proofs are formalizable in PA since all the inductions used are clearly first-order expressible (modulo an arithmetization of the syntax). Thus, our proof of the strong normalization for F_{at} is elementary. Can we claim that it is *finitistic*, i.e., is it formalizable in primitive recursive arithmetic?

Primitive recursive (or Skolem) arithmetic PRA is the quantifier-free system of arithmetic usually associated with *finitism* (the philosophically inclined reader can consult [11] for a defense of the position that finitistic reasoning is essentially the reasoning formalizable in PRA). The main features of PRA are the inclusion of a function symbol for each description of a primitive recursive function and, also, an appropriate rule of induction (see section 2.1 of [13] for a modern exposition of this system). The reader can readily object in two ways to the claim that we have provided

a finitistic proof of strong normalization. On the one hand, our arguments use first-order reasoning (and not quantifier-free inferences): even the very statement of strong normalization is – as it stands – not quantifier-free (it is given by a Π_2-sentence). On the other hand, many inductions involve *prima facie* Σ_1-predicates like '$t \in \mathbf{SN}$' or '$t \in \mathbf{FR}$' (hence, the inductions are not primitive recursive in character). These are two quite different objections, but they can be both answered by the following single result:

THEOREM 3 (Parsons, Takeuti, Mints). *If the theory* $\mathsf{I}\Sigma_1$ *proves a* Π_2*-sentence* $\forall x \exists y A(x, y)$*, where* A *is quantifier-free, then there is a (description for a) primitive recursive function* f *such that the theory* PRA *proves* $A(x, f(x))$*.*

The theory $\mathsf{I}\Sigma_1$ is the first-order subsystem of Peano arithmetic with induction restricted to Σ_1-formulas of arithmetic. It is well-known that $\mathsf{I}\Sigma_1$ is able to introduce (in an appropriate sense) all the primitive recursive functions (this is essentially a result of Gödel in his famous incompleteness paper). Hence, we can see PRA as a subtheory of $\mathsf{I}\Sigma_1$ (in the theorem above, we assume that the quantifier-free formula $A(x, y)$ is a formula of the language of PRA). Theorem 3 was proved, independently, by the referred authors in the early seventies (see [3] for references and a simple model-theoretic proof). In a nutshell, in order to see that a proof of a Π_2-statement is finitistic it is enough to be able to formalize it in the first-order theory $\mathsf{I}\Sigma_1$.

Are the proofs we present in this paper formalizable in $\mathsf{I}\Sigma_1$? An attentive checking of the proofs (which explains our care in making many steps and lemmas explicit) shows that all arguments can be formalized in $\mathsf{I}\Sigma_1$ except for a single one. It is the inductive argument of Lemma 2. Disregarding issues of type matching (to make reading easier), the following is proved:

$$\forall \rho \, (\forall r \in \mathbf{SN} \, \forall t^\rho \in \mathbf{SN} \, (rt \in \mathbf{SN} \wedge r[t/y] \in \mathbf{SN})).$$

We are being careless about the variable 'y'. We may just consider that we are quantifying over all free assumption variables of r (this is a bounded quantification and poses no problems for a finitistic proof). The above universal statement is proved by induction on the build-up of the type ρ (it is the *main induction* in the proof given in Section 3). However, the matrix $\forall r \in \mathbf{SN} \forall t^\rho \in \mathbf{SN} \, (rt \in \mathbf{SN} \wedge r[t/y] \in \mathbf{SN})$ is *prime facie* Π_2 and, therefore, the required induction is unavailable in $\mathsf{I}\Sigma_1$. Even though the

induction hypothesis (main induction hypothesis) is only used once – as it happens, in the seemingly innocent (var) case – it is used crucially there. Note that the problem is not intrinsic of atomic polymorphism, it is already present for the implicational fragment of the typed λ-calculus.

We do not see a way around this problem. It seems that the strategy in [8] for producing short proofs of normalization gives rise to elementary proofs only, not finitistic proofs. A way to transform elementary proofs like the ones we present in this paper, into finitistic ones is via an estimation of bounds. If we manage to bound primitive recursively the sizes of the **SN**-derivations and of the reduction trees, the predicates '$t \in \mathbf{SN}$' and '$t \in \mathbf{FR}$' become primitive recursive predicates. A natural attempt towards this goal is to try to adapt Helmut Schwichtenberg's strategy in [9] (for the implicational fragment of the typed λ-calculus) to the atomic polymorphic context (see also the refinement of Arnold Beckmann in [2]). We plan to pursue the delicate work of the estimation of upper bounds in future investigations. There are in the literature other approaches for the obtention of upper-bounds of the normalization procedure, as witnessed by the work of Jaco van de Pol [14]. However, this latter approach has some drawbacks since it analyses a proof of normalization based on Tait's computable predicate.

As pointed by an anonymous referee, other restrictions of system F with (limited) second-order quantification admit proofs of normalization with restricted means. We refer here the example of the referee, namely the work of Thorsten Altenkirch and Thierry Coquand [1] where the restriction occurs not in the instantiation of the universal quantifications but on the universal types allowed: no nesting of universal quantifiers is permited.

Acknowledgements

Both authors acknowledge the support of Fundação para a Ciência e a Tecnologia [UID/MAT/04561/2013] and Centro de Matemática, Aplicações Fundamentais e Investigação Operacional of Universidade de Lisboa. The second author is also grateful to Fundação para a Ciência e a Tecnologia [UID/CEC/00408/2013 and grant SFRH/BPD/93278/2013], to Large-Scale Informatics Systems Laboratory (Universidade de Lisboa) and to Núcleo de Investigação em Matemática (Universidade Lusófona).

References

[1] T. Altenkirch and T. Coquand, *A finitary subsystem of the polymorphic λ-calculus*, *Proceedings of the 5th International Conference on Typed Lambda Calculi and Applications (TLCA 2001)*, **Lecture Notes in Computer Science** 2044 (2001), pp. 22–28.

[2] A. Beckmann, *Exact bounds for lenghts of reductions in typed λ-calculus*, **The Journal of Symbolic Logic** 66(3) (2001), pp. 1277–1285.

[3] F. Ferreira, *A simple proof of Parsons' theorem*, **Notre Dame Journal of Formal Logic** 46 (2005), pp. 83–91.

[4] F. Ferreira, *Comments on predicative logic*, **Journal of Philosophical Logic** 35 (2006), pp. 1–8.

[5] F. Ferreira and G. Ferreira, *Atomic polymorphism*, **The Journal of Symbolic Logic** 78 (2013), pp. 260–274.

[6] F. Ferreira and G. Ferreira, *The faithfulness of* F_{at}*: a proof-theoretic proof*, **Studia Logica** 103(6) (2015), pp. 1303–1311.

[7] J.-Y. Girard, Y. Lafont and P. Taylor, **Proofs and Types**, Cambridge University Press (1989).

[8] F. Joachimski and R. Matthes, *Short proofs of normalization for the simply-typed lambda-calculus, permutative conversions and Gödel's T*, **Archive for Mathematical Logic** 42 (2003), pp. 59–87.

[9] H. Schwichtenberg, *An upper bound for reduction sequences in the typed λ-calculus*, **Archive for Mathematical Logic** 30 (1991), pp. 405–408.

[10] W. Tait, *Intentional interpretations of functionals of finite type I*, **The Journal of Symbolic Logic** 32 (1967), pp. 198–212.

[11] W. Tait, *Finitism*, **Journal of Philosophy** 78 (1981), pp. 524–546.

[12] A. S. Troelstra and H. Schwichtenberg, **Basic Proof Theory**, Cambridge University Press (1996).

[13] A. S. Troelstra and D. van Dalen, **Constructivism in Mathematics. An Introduction**, volume 1, North Holland, Amsterdam (1988).

[14] J. van de Pol, *Two different strong normalization proofs? Computability versus functionals of finite type*, *Proceedings of the Second International Workshop on Higher-Order Algebra, Logic and Term Rewriting (HOA'95)*, **Lecture Notes in Computer Science** 1074 (1996), pp. 201–220.

[15] F. van Raamsdonk and P. Severi, *On normalization*, Technical report CS-R9545, Centrum voor Wiskunde en Informatica, Amsterdam (1995).

Departamento de Matemática
Faculdade de Ciências, Universidade de Lisboa
Campo Grande, Ed. C6, 1749-016 Lisboa, Portugal
e-mail: fjferreira@fc.ul.pt

Departamento de Matemática
Faculdade de Ciências, Universidade de Lisboa
Campo Grande, Ed. C6, 1749-016 Lisboa, Portugal

and

Departamento de Matemática
Universidade Lusófona de Humanidades e Tecnologias
Av. do Campo Grande, 376, 1749-024
Lisboa, Portugal
e-mail: gmferreira@fc.ul.pt