M2 LMFI – SOFIX
Quantification du second-ordre et points fixes en logique
Expressiveness of System F

Alexis Saurin

February 2024

Representation theorem for F

The aim of the rest of the lecture is to prove the following theorem:

Theorem (1)

A function $f : \mathbb{N}^k \to \mathbb{N}$ *is representable in* F *is, and only if, it is provably total in* PA_2.

Every F-representable function is provably total in PA_2

Proposition (2)

If $f : \mathbb{N}^k \to \mathbb{N}$ *is representable in* F*, then is provably total in* PA_2*.*

Remark

SN of F *cannot be formalized in* PA_2*. Indeed, one would need to do an induction over* **reducibility relations** *which are parametrized by a* **valuation of type variables to reducibility candidates**. *Valuations* v *can be represented as sets of natural numbers, but not by natural numbers (as* $\{n \in \mathbb{N}/\mathrm{r}(n) \in \mathrm{v}(\mathrm{I}(\mathrm{n}))\}$*).*
SN expresses as: for any $t : T$ *and for any valuation* v*, if* $t \in \mathrm{Red}_{\mathrm{v}}(T)$*, then* t *is SN.* **This is above 2nd order.**

Proof: To type $(t)\overline{n_1} \ldots \overline{n_k}$, one needs **only a finite set of types** of F, say $\{T_1, \ldots, T_k\}$ which is closed by subformula.
Define n predicates $\mathrm{Red}_i(n, E)$ expressing the fact that the term of Gödel number n is reducible of type T_i in the valuation represented by E.
One does not need to perform an induction but simply to define those k formulas in the appropriate order: no need anymore to quantify over valuations. The strong normalization of **any particular** F term can be formalized in PA_2. □

The difficulty of the converse

In the following, one shall now focus on the converse property: functions which are provably total in PA_2 are F-representable.
The difficulty comes from the fact that PA_2 contains a lot of features that are not present in system F:

- richness of the language of formulas, while F is built from universal quantification and implication only
- classical reasoning, while F is intuitionistic,
- first-order statements, while F is propositional,
- arithmetical axioms while F is purely logical (axiom-free)

In order to prove the converse, one will step by step define more and more restricted versions of arithmetic (resp. as well as extensions of system F) which preserve the class of provably recursive functions (resp. of representable functions)

A specialization of Peano's fourth axiom

The first step consists in modifying Peano's fourth axiom to allow for a better compatibility with F.

Definition

The theory PA_2' is obtained from PA_2 by replacing axiom (P_4) by

$$(P_4') \qquad \forall x.\forall y.(sx = 0 \Rightarrow y = 0).$$

Lemma

If a function is provably total in PA_2 *then it is also provably total in* PA_2'.

Proof: Let ϕ denote the formula $\forall x.\exists y.t_f(x, y) = 0$ and suppose that ϕ has a proof in PA_2. It is not hard to see that from (P_4') one can (classically) derive the disjunction $P_4 \vee \forall y(y = 0)$.
From either of the components we can derive ϕ. Indeed, with (P_4) we regain the power of the full PA_2, and in the other case everything is zero anyway.
Therefore ϕ has a proof in PA_2'. □

Heyting and Peano second-order arithmetics: dealing with classical reasoning

When $k(\cdot)$ denotes Kolmogorov double negation translation (which introduces double negations $\neg\neg\cdot$ in front of every connective and atomic formula) one has the following:

Proposition (6)

For all ϕ, $\mathsf{PA}_2 \vdash \phi$ *is equivalent to* $\mathsf{HA}_2 \vdash k(\phi)$ *and* $\mathsf{PA}_2' \vdash \phi$ *is equivalent to* $\mathsf{HA}_2' \vdash k(\phi)$.

Definition (Friedman's translation)

Let ρ be a fixed formula. We denote ϕ^ρ the formula obtained from ϕ by replacing every atomic subformula A (including $\bot$) by the disjunction $A \vee \rho$. (It is assumed that free variables of ρ are not bound in ϕ.) That is:

- $(X(t_1, \ldots, t_n)^\rho = X(t_1, \ldots, t_n) \vee \rho$;
- $(\bot)^\rho = \bot \vee \rho$;
- $(A \alpha B)^\rho = A^\rho \alpha B^\rho$, for $\alpha \in \{\vee, \wedge, \Rightarrow\}$;
- $(Qx.A)^\rho = Qx.A^\rho$, for $Q \in \{\forall^1, \exists^1\}$;

Lemma (8)

We have the following properties (ρ is the formula used in Friedman's translation):

1. *For every ϕ, the formula $\vdash \rho \Rightarrow \phi^\rho$ is provable in NJ2;*
2. *For every ϕ and t, if $x \notin FV(\rho)$ then $(\phi[t/x])^\rho = \phi^\rho[t/x]$;*
3. *Let X be a n-ary second-order variable, $\vec{a}$ by n distinct first-order variables such that $X, \vec{a} \notin FV(\rho)$ and ψ be a formula. Then, for every ϕ, $(\phi[\psi/X(\vec{a})])^\rho \Leftrightarrow \phi^\rho[\phi^\rho/X(\vec{a})]$ is provable in NJ2;*
4. *If $\Gamma \vdash \phi$ then $\Gamma^\rho \vdash \phi^\rho$, where $\Gamma^\rho = \{\gamma^\rho \mid \gamma \in \Gamma\}$.*

Proof: The proof of parts (i) and (ii) is by a routine induction with respect to ϕ.
Both (i) and (ii) are used in the proof of (iii), which is also an easy induction on ϕ.
Part (iv) goes by induction with respect to $\Gamma \vdash \phi$, using parts (ii) and (iii) in cases when the proof involves instantiation. □

Lemma

The following holds:

(i) *If* ϕ *is an axiom of* HA_2' *then* $\mathsf{HA}_2' \vdash \phi^\rho$.

(ii) *If* $\mathsf{HA}_2' \vdash \phi$ *then* $\mathsf{HA}_2' \vdash \phi^\rho$.

Proof: For part (i) observe that most of the axioms are universal formulas of the form $\phi = \forall \vec{x}(\alpha_1 \Rightarrow \cdots \Rightarrow \alpha_n \Rightarrow \beta)$, where α_i and β are equations. Then ϕ^ρ is $\forall \vec{x}((\alpha_1 \vee \rho) \Rightarrow \cdots \Rightarrow (\alpha_n \vee \rho) \Rightarrow (\beta \vee \rho))$ and this formula is easily provable from ϕ.
The only axiom not of the form above is (*Ind*), but its translation, Ind^ρ is essentially a special case of (*Ind*) when quantification is specialized to range only on formulas of the form $X \vee \rho$.

Part (ii) is an immediate consequence of part (i) and Lemma 8.(iv). □

Finally we can prove that PA_2 and HA_2' have the same provably total functions:

Theorem (10, Spector)

If a function f is provably total in PA_2 then it is also provably total in HA_2'.

Proof: Given f provably total in PA_2, we know that it is provably total in PA_2'. Thus PA_2' proves $\exists y.t_f(\vec{n}, y) = 0$.
Using double negation translation, (Lemma 6), we have
$HA_2' \vdash \neg\neg\exists y.\neg\neg t_f(\vec{n}, y) = 0$.
To this formula we apply Friedman's translation, where we take ρ to be $\exists y.t_f(\vec{n}, y) = 0$, and (after simplifying $\bot \vee \rho$ to ρ) we obtain:
$HA_2' \vdash ((\exists y.[((t_f(\vec{n}, b) = 0 \vee \rho) \Rightarrow \rho) \Rightarrow \rho] \Rightarrow \rho) \Rightarrow \rho$.
The formula $(t_f(\vec{n}, b) = 0 \vee \rho) \Rightarrow \rho$ is provable, and thus
$HA_2' \vdash ((\exists y.\rho) \Rightarrow \rho) \Rightarrow \rho$.
But of course $\exists y.\rho$ is equivalent to ρ, so we obtain $HA_2' \vdash (\rho \Rightarrow \rho) \Rightarrow \rho$ and finally we have $HA_2' \vdash \rho$. □
Note that the converse is trivial since PA_2 is stronger than PA_2'.

A restricted syntax, HA_2^-

In the following we shall restrict Heyting arithmetic to a core, minimal syntax, removing functions symbols, etc. Every primitive recursive function can be defined in arithmetic, so one can get rid of the primitive function recursive symbols as long as one has zero and successor symbols:

Lemma

For every primitive recursive f of k arguments there exists a formula ϕ_f with $k+1$ free individual variables, and with no symbols of primitive recursive function, such that $\mathrm{HA}_2' \vdash f(\vec{n}) = m \Leftrightarrow \phi_f(\vec{n}, m)$.

Note also that all logical connectives were definables in terms of $\Rightarrow$ and $\forall_1, \forall_2$, as well as equality which can be taken as Leibniz equality.
In the following, one shall restrict to this core syntax, referred to as HA_2^-.

Definition

Let ϕ be a formula in the language of HA_2'. By ϕ^- we denote the formula (over the syntax of HA_2^-) obtained from ϕ by replacing all occurrences of $\wedge, \vee, \exists_1, \exists_2, =$ by their second-order definitions. The notation Γ^- (in particular HA_2^-) is used accordingly for sets of formulas.

Lemma

If $\Gamma \vdash \phi$ *then* $\Gamma_L \vdash \phi_L$.

HA_2^- can be restricted to (with $=_L$ being the Leibniz equality):

- $P_3 = \forall x, y.(Sx =_L Sy \Rightarrow x =_L y)$;
- $P_4' = \forall x, y(Sx =_L 0 \Rightarrow y =_L 0)$;
- $P_5^- = \forall x.\mathrm{Nat}(x)$.

Then, we can even get rid of P_5 by relativizing all first-order universal quantifications: we abbreviate $\forall x.(\mathrm{Nat}(x) \Rightarrow F)$ as $\forall x^{\mathrm{Nat}}.F$.

Definition

If ϕ is in the simplified syntax, ϕ^{Nat} is obtained by relativizing all quantifiers of ϕ . If Γ is a set of formulas in the simplified syntax then Γ^{Nat} is obtained by:

- relativizing all formulas in Γ to Nat;
- adding assumptions $\mathrm{Nat}(x)$, for all $x \in FV(\Gamma)$;
- adding the axioms P_3 and P_4'.

$\mathrm{Nat}(\vec{x})$ abbreviates the conjunction of $\mathrm{Nat}(x)$ for $x \in \vec{x}$.

Note that $(\exists x.\phi)^{\mathrm{Nat}} = \forall X.((\forall x(\mathrm{Nat}(x) \Rightarrow \phi^{\mathrm{Nat}}) \Rightarrow X) \Rightarrow X)$, which is the formula as $\exists x.(\mathrm{Nat}(x) \wedge \phi^{\mathrm{Nat}})$, so that relativization commutes with removing definable connectives.

Lemma (15)

If ϕ is a formula in the simplified syntax:

1. $\mathsf{HA}_2^- \vdash \phi^{\mathsf{Nat}} \Leftrightarrow \phi$;
2. $(\phi[t/x])^{\mathsf{Nat}} \Leftrightarrow \phi^{\mathsf{Nat}}[t/x]$;
3. $(\phi[F/X(x_1, \ldots, x_n)])^{\mathsf{Nat}} \Leftrightarrow \phi^{\mathsf{Nat}}[F^{\mathsf{Nat}}/X(x_1, \ldots, x_n)]$;
4. *If* $\Gamma \vdash \mathsf{Nat}(x) \Rightarrow \phi$ *and* $x \notin FV(\Gamma, \phi)$, *then* $\Gamma \vdash \phi$.

Proof: Parts (1-3) are shown by induction with respect to ϕ.
To show part (4), generalize over x and instantiate it to 0. □

Lemma

Let ϕ be a formula in the simplified syntax with $FV(\phi) = \vec{x}$. Then $\mathsf{HA}_2^-, \Gamma \vdash \phi$ if and only if $\Gamma^{\mathsf{Nat}}, \mathsf{Nat}(\vec{x}) \vdash \phi^{\mathsf{Nat}}$.

Proof: The proof from left to right is by induction with respect to the derivation of ϕ:
Among the axioms of HA_2^- only induction is non-obvious.
We need to find a derivation for the following judgement:

$$\Gamma^{\mathsf{Nat}}, \mathsf{Nat}(x) \vdash \forall X.(\forall y.(\mathsf{Nat}(y) \Rightarrow (X(y) \Rightarrow X(S(y))) \Rightarrow (X(0) \Rightarrow \forall y.X(y)).$$

This can be done by instantiating the $\forall_2 X$ in $\mathsf{Nat}(x)$ by $\mathsf{Nat}(y) \wedge X(y)$.
The other cases in the proof are routine, but Lemma 15 is needed.

For the other direction we simply use the first case of Lemma 15:
Since HA_2^-, Γ proves all formulas in $\Gamma^{\mathsf{Nat}}, \mathsf{Nat}(\vec{x})$, we have $\mathsf{HA}_2^-, \Gamma \vdash \phi^{\mathsf{Nat}}$ and thus $\mathsf{HA}_2^-, \Gamma \vdash \phi$. □
As a conclusion:

Lemma

If ϕ is closed and $\mathsf{HA}_2' \vdash \phi$ then $P_3^-, P_4' \vdash \phi^{-\mathsf{Nat}}$.

We shall now consider a slight extension of the system F that we having been working with until now:

In addition to Church-style System F, we extend the grammar of types with first-order quantifications and terms:

$$\frac{\Gamma \vdash_{\mathsf{F}} M : \forall a.A}{\Gamma \vdash_{\mathsf{F}} (M)t : A[t/a]} \qquad \frac{\Gamma \vdash_{\mathsf{F}} M : A}{\Gamma \vdash_{\mathsf{F}} \lambda a.M : \forall a.A}$$

(If $a \notin FV(A)$ in the second rule.) and we consider two constants: p_3, p_4:

- $\vdash_{\mathsf{F}} p_3 : \forall a, b.(S(a) =_L S(b) \Rightarrow a =_L b)$;
- $\vdash_{\mathsf{F}} p_4 : \forall a, b.(S(a) =_L 0 \Rightarrow b =_L 0)$.

Since we are now modelling not only propositional second-order logic but full second-oder logic, universal abstraction will get more expression and we shall also slightly extend the notation for types since they must now also describe n-ary relations. For this, given a formula ϕ and n-variables $\vec{a}$, we will form type $\lambda\vec{a}.\phi$ which will allow us to express the full second-order elimination rule as

$$(M)\lambda\vec{a}.\phi.$$

Lemma (18)

1. *If a closed formula ϕ is a theorem of HA_2' then ϕ^{Nat} is an inhabited type of system* F.
2. *If a type ϕ^{Nat} is inhabited in* F *then ϕ is (classically) true in the standard model of arithmetic.*

Proof: Part (1) is an immediate consequence of the previous lemma. For part (2), one checks that the axioms are true in the standard model $\mathcal{N}$ and that $\mathcal{N} \models \phi \Leftrightarrow \phi^{\mathsf{Nat}}$. □

Let us turn to consider the reduction associated with this extended System F:

The reduction relation of F is extended with:

$$(\lambda a.M)t \longrightarrow M[t/a]$$

and universal reduction shall now take into account that type variables may have a non-zero arity:

$$(\Lambda X.M)\lambda\vec{a}.\phi \longrightarrow M[\phi/X(a)]$$

Regarding p_3, if t is a first-order term and M is a term of type $S(t) =_L S(t)$, $(p_3)ttM$ is a term of type $t =_L t$ which is $\forall X.X(t) \Rightarrow X(t)$. We therefore add a specific rule for p_3:

$$(p_3)ttM \longrightarrow \Lambda X.\lambda x^{X(t)}.x.$$

On the other hand, we do not introduce additional reduction for p_4.

The main properties of $\lambda\mathsf{HA}_2$ are the following:

Theorem

1. $\lambda\mathsf{HA}_2$ *is weakly confluent;*
2. $\lambda\mathsf{HA}_2$ *is strongly normalizing;*
3. $\lambda\mathsf{HA}_2$ *is Church-Rosser.*

Weak confluence derives naturally from the absence of critical pairs.

Strong normalization follows from an embedding into F

Definition

The contracting map b maps terms of system $\lambda \mathrm{HA}_2$ to terms of system F, where $\mathrm{ID} = \forall X.(X \to X)$. For types:
$\mathrm{b}(R(t_1, \ldots, t_n)) = V_R$, where V_R is the propositional variable associated to R by an injection from second-order relational symbols to system F propositional second-order variables.
$\mathrm{b}(\forall X.T) = \forall X.\mathrm{b}(T)$; $\mathrm{b}(T \Rightarrow U) = \mathrm{b}(T) \to \mathrm{b}(U)$;
For terms:

- $\mathrm{b}(x) = x$;
- $\mathrm{b}(p_3) = \mathrm{b}(p_4) = \lambda x^{\mathrm{ID}}.x$;
- $\mathrm{b}(\lambda x^T.M) = \lambda x^{\mathrm{b}(T)}.\mathrm{b}(M)$;
- $\mathrm{b}(\lambda a.M) = \mathrm{b}(M)$;
- $\mathrm{b}(\Lambda X.M) = \Lambda V_X.\mathrm{b}(M)$;
- $\mathrm{b}((M)N) = (\mathrm{b}(M))\mathrm{b}(N)$;
- $\mathrm{b}((M)t) = \mathrm{b}(M)$;
- $\mathrm{b}((M)\lambda \vec{a}.\phi) = (\mathrm{b}(M))\mathrm{b}(\phi)$.

Lemma

The operation b *commutes with substitution, i.e, the following equations always hold;*

- $\mathsf{b}(\phi[t/a]) = \mathsf{b}(\phi)$
- $\mathsf{b}(\phi[\lambda\vec{a}.\phi/X]) = \mathsf{b}(\phi)[\mathsf{b}(\phi)/V_X]$;
- $\mathsf{b}(M[t/a]) = \mathsf{b}(M)$;
- $\mathsf{b}(M[N/x]) = \mathsf{b}(M)[\mathsf{b}(N)/x]$;
- $\mathsf{b}(M[\lambda\vec{a}.\phi/R]) = \mathsf{b}(M)[\mathsf{b}(\phi)/V_R]$.

Lemma

If $\Gamma \vdash_{\lambda HA_2} M : \phi$, *then* $b(\Gamma) \vdash_F b(M) : b(\phi)$.

Lemma (23)

If $M \longrightarrow_{\lambda HA_2} N$ *then* $b(M) \longrightarrow_F^{\star} b(N)$ *In addition,* $b(M) \longrightarrow_F^{+} b(N)$ *in all cases except when the reduction step* $M \longrightarrow_{\lambda HA_2} N$ *is an application of first-order beta-reduction.*

Proof: It is simply a matter of a check of each reduction rule. □

Proof:[of strong normalization] Strong normalization of λHA_2 simply follows from that of F as there cannot be an infinite sequence of first-order reductions. □

Proof:[of Church-Rosser] It is a simple application of Newman's lemma from weak confluence and strong normalization of λHA_2. □

Definition (Target variable)

The ***target variable*** of a formula ϕ (in the simplified syntax) is the rightmost relation variable occurring in ϕ:

- R is the target of $R(t_1, \ldots, t_n)$
- the target of $\forall_1 a.\phi$, $\forall_2 R.\phi$, $\psi \Rightarrow \phi$ is the target of ϕ.

We say that an environment Γ is ***easy*** if the target of every $T \in \Gamma$ is free in Γ.

Observe that if Γ is easy, $(x : T) \in \Gamma$ and $\Gamma \vdash_{\lambda \mathsf{HA_2}} (x)M : U$, then T and U have the same target variable.

Lemma (25)

1. *If M is normal then it is either an abstraction or an application $(c)\vec{e}$ (c either constant or a variable, and $\vec{e}$ is a (possibly empty) sequence of terms and types).*
2. *If $(c)\vec{e}$ is normal beginning with a constant with no free first-order variables, its type in an easy environment can only be of the form;*
   - $\forall a.\forall b.(S(a) = S(b) \to a = b)$;
   - $\forall vx.(S(\underline{n}) = S(a) \to \underline{n} = a)$.
   - $S(\underline{n}) = S(\underline{m}) \to \underline{n} = \underline{m}$;
   - $\forall a, b.(S(a) = 0 \to b = 0)$;
   - $\forall b.(S(\underline{n}) = 0 \to b = 0)$;
   - $S(\underline{n}) = 0 \to \underline{m} = 0$.
3. *A normal form without free individual variables, which has type $\underline{n} = \underline{m}$ in an easy environment must be of shape $\Lambda X.\lambda x^{X(\underline{n})}.x$. In particular, if $\underline{n} = \underline{m}$ is inhabited then $n = m$.*
4. *Let $\Gamma, x : \forall a.(F \to (G \to X)) \vdash M : \mathrm{Nat}(\underline{n})$, where Γ is easy, X is a nullary relation variable, and M is a normal form without free first-order variables, and let $O = \lambda x^{\omega}.\Lambda X.\lambda y^{X\to X}.\lambda z^{X}.(x)Xyz$. Then $(O)(\mathrm{b}(M)) =_{\beta} \overline{n}$.*

Proof: Part (1) is shown by induction with respect to M. If M is neither a constant nor an abstraction, then it must be an application $(M_1)e$. and we can apply the induction hypothesis to M_1. But now M_1 cannot be an abstraction, because $(M_1)e$ would be a redex.
Parts (2) and (3) are shown by parallel induction with respect to the size of terms. For part (ii) observe that in every application of the form $(p_3)nmP$, the term P has type $S(n) = S(m)$. By the induction hypothesis, part (iii), we have $n = m$, and thus our term is a redex, which is a contradiction. Similarly, an application $(p_4)nmP$ would contain a closed term P of type $S(n) = 0$ and this is again impossible by the induction hypothesis (iii).
For part (3) observe that, by part (2), a normal form of type $n = m$ cannot be an application beginning with a constant. It cannot be an application beginning with a variable, because the target of $n = m$ is not free. It canoot be a variable so it is an abstraction. □

Proof:

For part (4), observe that $\Gamma, x : \forall a(T \to S \to X) \vdash M : \mathrm{Nat}(n)$ implies, by parts (i) and (ii), that $M = \Lambda R.\lambda y^{\forall a.(R(a) \to R(S(a)))}.N$ where N has type $R(0) \to R(n)$ in the environment $\Gamma' = \Gamma, x : \forall a.(R(a) \to R(S(a)))$. We can also assume that $R \notin FV(\Gamma)$. Then either $n = 1$ and $N = (y)O$, or $N = \lambda z^{R(0)} N'$ and we have $\Gamma', z : R(0) \vdash N' : R(n)$. In the latter case, by induction with respect to the size of N', it follows that $\mathrm{b}(N) = (y)^n z$. Indeed, by parts (i) and (ii), either $N' = z$ (in which case $n = 0$), or $N' = (y)\underline{m}N''$ with N'' of type $R(m)$ (and then $n = m + 1$). We conclude that $\mathrm{b}(M)$ is either c_n or $\Lambda X.\lambda y^{X \to X}.y$ (and $n = 1$). In each case $(O)\mathrm{b}(M) = c_n$. □

Theorem (27, Girard)

All functions provably total in PA_2 *are definable in F.*

Proof: If f is provably total in PA_2, then by Theorem 10 it is also provably total in HA_2', Without loss of generality (Exercise 12.15) we may assume that f is unary.
By Lemma 18 there exists a closed term M of $\lambda\mathrm{HA}_2$ of the following type (where $t_f(\underline{n}, b) = 0$ is an abbreviation):

$$\forall a(\mathrm{Nat}(a) \to \forall_2 R(\forall b.(\mathrm{Nat}(b) \to (t_f(a,b) = 0)^{\mathrm{Nat}} \to R) \to R)).$$

Let $n \in \mathbb{N}$ and let N_n be a closed term of type $\mathrm{Nat}(n)$, such that $\mathrm{b}(N_n) = c_n$.
The application $(M)nN_n$ is a closed term of type

$$\forall_2 R(\forall b(\mathrm{Nat}(b) \to (t_f(n,b) = 0)^{\mathrm{Nat}} \to R) \to R).$$

By Lemma 25.(1-2), this term reduces to

$$\Lambda X.\lambda x^{\forall b.(\mathrm{Nat}(b) \to (t_f(\underline{n},b)=0)^{\mathbf{Nat}} \to X)}.(x)\underline{m}NN',$$

for some m, where

1. $x : \forall b(\mathrm{Nat}(b) \to (t_f(n,b) = 0)^{\mathrm{Nat}} \to X) \vdash N : \mathrm{Nat}(\underline{m}$;
2. $x : \forall b(\mathrm{Nat}(b) \to (t_f(n,b) = 0)^{\mathrm{Nat}} \to X) \vdash N' : (t_f(n,m) = 0)^{\mathrm{Nat}}$.

Theorem (27, Girard)

All functions provably total in PA_2 *are definable in F.*

Proof: (continued)
From (2) it follows that

$$\vdash N'[(P \to P)/X][\lambda b.\lambda y^{\mathrm{Nat}(b)}.\lambda z^{t_f(n,b)=0)^{\mathbf{Nat}}}.\lambda v^{P}.v/x] : (t_f(n,m) = 0)^{\mathrm{Nat}},$$

where P is any nullary relation variable. That is, the type $(t_f(n,m) = 0)^{\mathrm{Nat}}$ is inhabited in $\lambda\mathrm{HA}_2$, and thus $\mathcal{N} \models t_f(\underline{n},\underline{m}) = 0$, by Lemma 25.(2). It follows that $f(n) = \mathsf{I}(m)$, where I is the left converse to the pairing function.
Let $S = \mathrm{b}((t_f(n,b) = 0)^{\mathrm{Nat}})$. Then f is definable in system F by the term

$$F = \lambda n^{\omega}.(L)(O)(\mathrm{b}(M)n\omega\lambda m^{\omega}.\lambda y^{T}.m),$$

where L is a term representing I. Indeed, by inspecting the reduction of $(M)nN_n$, and applying Lemma 23, we obtain that the term $\mathrm{b}(M)c_n$ reduces to $\Lambda X.\lambda x^{\omega\to S\to X}.(x)\mathrm{b}(N)\mathrm{b}(N')$. It follows that $\mathrm{b}(M)c_n\omega(\lambda m^{\omega}.\lambda y^{S}.m)$ reduces to $\mathrm{b}(N)$. From (1) and Lemma 25(iv) we have $(O)\mathrm{b}(N) = c_m$. Thus, the application $(F)c_n$ reduces to $L(c_m)$, where m is such that $f(n) = \mathsf{I}(m)$. □

Corollary

The class of functions definable in F coincides with the class of provably recursive functions of second-order Peano Arithmetic.

We already know that product, booleans and natural numbers can be represented as typed terms on System F.
To embed T in F, it is sufficient to represent the higher-order recursor of T, which is done simply as follows:

$$\mathsf{Rec} \triangleq \lambda n^{\mathsf{Nat}}.\lambda f^{\mathsf{Nat}\to(U\to U)}.\lambda b^{U}.(\pi_1)(n)(U\times\mathsf{Nat})\langle b,\overline{0}\rangle\lambda z^{U\times\mathsf{Nat}}.\langle((f)(\pi_2)z)(\pi_1)z$$