Monotone Fixed-Point Types and Strong Normalization

Ralph Matthes*

Institut für Informatik
Ludwig-Maximilians-Universität München
Oettingenstraße 67, D-80538 München, Germany
matthes@informatik.uni-muenchen.de

Abstract. Several systems of fixed-point types (also called retract types or recursive types with explicit isomorphisms) extending system F are considered. The seemingly strongest systems have monotonicity witnesses and use them in the definition of beta reduction for those types. A more naïve approach leads to non-normalizing terms. All the other systems are strongly normalizing because they embed in a reduction-preserving way into the system of non-interleaved positive fixed-point types which can be shown to be strongly normalizing by an easy extension of some proof of strong normalization for system F.

1 Introduction

It is well-known that inductive data types may be encoded impredicatively. Unfortunately the encoding in system F only models iteration. In [5] it is shown for the example Int of natural numbers that full primitive recursion is only achieved for numerals instead of arbitrary terms of type Int. We read (on p. 91): "We make no secret of the fact that this is a defect of system F." The second problem (also shown in [5]) with this encoding is the lack of efficiency: The predecessor function (which only works on numerals) needs $O(n)$ steps to compute the predecessor of the successor of numeral n.

It is now tempting to study extensions of system F by inductive types with full primitive recursion overcoming those defects ([4], [9]). However, led by the fact that those anomalies of the encoding already show up for the predecessor function, we study extensions by fixed-point types which amounts to postulating the existence of efficient[1] predecessors for the data types under consideration.

In [4] such an extension (by "retract types") is studied for "positive type schemes". In the present paper we free ourselves from positivity restrictions and replace them by monotonicity witnesses which are terms in the system. Even then a naive definition leads to non-normalizing terms as shown by Lemma 1.

* I am thankful for support by the Volkswagen-Stiftung.

[1] An efficient predecessor computes the predecessor of a successor term in a constant number of steps.

This problem is overcome by a beta rule with a somewhat artificial reference to the monotonicity witness (Theorem 1).

In sections 3 and 4 we study positivity restrictions (slightly more general than in [4]) and verify that they give rise to monotonicity witnesses which are well-behaved in the sense that the above artificial reference is compensated with. The key property is the first functor law stating that the identity is mapped to the identity (the proof of which requires some extensionality expressed by eta rules). Therefore we may say that the new beta rule for μ helps to separate concerns: Strong normalization is guaranteed for any monotone fixed-point type and an efficient predecessor is achieved if we find a monotonicity witness obeying the first functor law.

In section 5 a streamlined proof of strong normalization following the saturated sets approach is outlined. It turns out that most of the work is done for system F and that the extra clauses for non-interleaving positive $\mu\alpha\rho$ are dealt with very easily.

Strong normalization of the systems of monotone fixed-point types now follows from the embeddings shown in section 6.

2 Definition of Systems of Monotone Fixed-Point Types

We consider extensions of system F (see e. g. [5]) by fixed-point types, i. e. we have infinitely many type variables (denoted by α, β, ...) and with types ρ and σ we also have the product type $\rho \times \sigma$ and the function type $\rho \to \sigma$. Moreover, given a variable α and a type ρ we form the universal type $\forall\alpha\rho$ and the fixed-point type $\mu\alpha\rho$. (The μ shall not indicate that we model a least fixed point. For ease of presentation we do not use another symbol.) The $\forall$ and μ bind α in ρ. The renaming convention for bound variables is adopted. Let $\mathsf{FV}(\rho)$ be the set of type variables occurring free in ρ.

The terms of F are presented as in [5], i. e. without contexts and with fixed types (see [3] p. 159 for comments on this original typing à la Church). We have infinitely many term variables with types (denoted e. g. by x^ρ), pairing $\langle r^\rho, s^\sigma\rangle^{\rho\times\sigma}$, projections $(r^{\rho\times\sigma}\mathsf{L})^\rho$ and $(r^{\rho\times\sigma}\mathsf{R})^\sigma$, lambda abstraction $(\lambda x^\rho r^\sigma)^{\rho\to\sigma}$ for terms, term application $(r^{\rho\to\sigma}s^\rho)^\sigma$, lambda abstraction $(\Lambda\alpha r^\rho)^{\forall\alpha\rho}$ for types (under the usual proviso) and type application $(r^{\forall\alpha\rho}\sigma)^{\rho[\alpha:=\sigma]}$. We freely use the renaming convention for bound term and type variables of terms.

Beta reduction $\mapsto$ for system F is as usual given by

$$
\begin{array}{lr}
(\beta_\times) & \langle r, s\rangle\mathsf{L} \mapsto r \\
 & \langle r, s\rangle\mathsf{R} \mapsto s \\
(\beta_\to) & (\lambda x^\rho r)s \mapsto r[x^\rho := s] \\
(\beta_\forall) & (\Lambda\alpha r)\sigma \mapsto r[\alpha := \sigma] \ .
\end{array}
$$

The reduction relation $\to$ is defined as the term closure of $\mapsto$. The main theorem on F (due to Girard) is that $\to$ is strongly normalizing, i. e. there are no infinite reduction sequences $r_1 \to r_2 \to r_3 \to \cdots$ (In [9] all the definitions and lemmas are given quite carefully so as to make sure that the main theorem can also be proved in this presentation without contexts.)

We now want to extend the term system of F by rules pertaining to the type construct $\mu\alpha\rho$ and also extend the notion of beta reduction without giving up strong normalization.

In presentations not concerned about normalization as e. g. [1] (the authors call the fixed-point types "recursive types with explicit isomorphism") there is a term construction which "folds" $\rho[\alpha := \mu\alpha\rho]$ into $\mu\alpha\rho$ and one which "unfolds" terms of type $\mu\alpha\rho$ to terms of type $\rho[\alpha := \mu\alpha\rho]$. This would give the rules:

(μ-I) If $t^{\rho[\alpha:=\mu\alpha\rho]}$ is a term, then $(\mathrm{C}_{\mu\alpha\rho}t)^{\mu\alpha\rho}$ is a term.
(μ-E) If $r^{\mu\alpha\rho}$ is a term, then $(r\mathrm{E}_\mu)^{\rho[\alpha:=\mu\alpha\rho]}$ is a term.

The obvious beta rule for μ would be

$$(\beta_\mu) \qquad (\mathrm{C}_{\mu\alpha\rho}t)\mathrm{E}_\mu \mapsto t \ .$$

[The presentation follows the pattern of natural deduction very closely: There is an introduction rule (μ-I) by which terms of the new type are introduced ("constructed"—hence the symbol $\mathrm{C}_{\mu\alpha\rho}$) and an elimination rule (μ-E) by which terms of the new type are used. A beta rule gives a reduct for a term in which an elimination follows an introduction.]

Because of this rule and its apparent similarity to retracts in domain theory the fixed-point types get the name retract types in [4].

Unfortunately this allows for non-normalizing terms which will be shown even for a restricted system in Lemma 1.

Having a normalization proof by saturated sets (some variant of Girard's candidates) in mind, and also Tarski's theorem that every monotone endofunction on a complete lattice has a fixed point, and that saturated sets form a complete lattice with set inclusion as partial ordering, the following restriction of the use of $\mu\alpha\rho$ comes very naturally: We may only use $\mu\alpha\rho$ if there is a term witnessing the monotonicity of $\lambda\alpha\rho$, i. e. if there is a term m of type $\forall\alpha\forall\beta.(\alpha \to \beta) \to \rho \to \rho[\alpha := \beta]$ (with $\beta \notin \{\alpha\} \cup \mathsf{FV}(\rho)$). We may enforce this restriction in two ways: Either by assuming the term m in the introduction rule (which will give the system $\mathsf{EMF}_{\mathsf{naive}}$ of elimination-based monotone fixed-point types) or by assuming it in the elimination rule (thus defining the system $\mathsf{IMF}_{\mathsf{naive}}$ of introduction-based monotone fixed-point types).

Hence, the terms of $\mathsf{EMF}_{\mathsf{naive}}$ are defined by extending the definition of terms of system F by

(μ-I) If $t^{\rho[\alpha:=\mu\alpha\rho]}$ is a term and $m^{\forall\alpha\forall\beta.(\alpha\to\beta)\to\rho\to\rho[\alpha:=\beta]}$ is a term, then $(\mathrm{C}_{\mu\alpha\rho}mt)^{\mu\alpha\rho}$ is a term.
(μ-E) If $r^{\mu\alpha\rho}$ is a term, then $(r\mathrm{E}_\mu)^{\rho[\alpha:=\mu\alpha\rho]}$ is a term.

The relation $\mapsto$ is extended for $\mathsf{EMF}_{\mathsf{naive}}$ by

$$(\beta_\mu) \qquad (\mathrm{C}_{\mu\alpha\rho}mt)\mathrm{E}_\mu \mapsto t \ .$$

Similarly, the terms of $\mathsf{IMF}_{\mathsf{naive}}$ are defined by extending the definition of terms of system F by

(μ-I) If $t^{\rho[\alpha:=\mu\alpha\rho]}$ is a term, then $(\mathrm{C}_{\mu\alpha\rho}t)^{\mu\alpha\rho}$ is a term.

(μ-E) If $r^{\mu\alpha\rho}$ and $m^{\forall\alpha\forall\beta.(\alpha\to\beta)\to\rho\to\rho[\alpha:=\beta]}$ are terms, then $(r\mathrm{E}_\mu m)^{\rho[\alpha:=\mu\alpha\rho]}$ is a term.

The relation $\mapsto$ of system F is extended for $\mathsf{IMF}_{\mathsf{naive}}$ by

$$(\beta_\mu) \qquad (\mathrm{C}_{\mu\alpha\rho}t)\mathrm{E}_\mu m \mapsto t \ .$$

Lemma 1. $\mathsf{EMF}_{\mathsf{naive}}$ *and* $\mathsf{IMF}_{\mathsf{naive}}$ *are not normalizing.*

Proof. We only show it for $\mathsf{EMF}_{\mathsf{naive}}$ because the proof for $\mathsf{IMF}_{\mathsf{naive}}$ is essentially the same. Set $1 := \forall\alpha.\alpha \to \alpha$, $\mathsf{IN1} := \Lambda\alpha\lambda x^\alpha.x$ (of type 1) and $\mathsf{p} := \mu\alpha.\alpha \to 1$. We get the trivial closed monotonicity witness $m := \Lambda\alpha\Lambda\beta\lambda f^{\alpha\to\beta}\lambda x^{\alpha\to 1}\lambda y^\beta.\mathsf{IN1}$ for $\lambda\alpha.\alpha \to 1$. Define $\omega := \lambda x^{\mathsf{p}}.(x\mathrm{E}_\mu)x$ of type $\mathsf{p} \to 1$. Then $C_{\mathsf{p}}m\omega : \mathsf{p}$ and therefore $\Omega := \omega(C_{\mathsf{p}}m\omega) : 1$. We calculate

$$\Omega \mapsto_{\beta_\to} (\mathrm{C}_{\mathsf{p}}m\omega)\mathrm{E}_\mu(\mathrm{C}_{\mathsf{p}}m\omega) \to_{\beta_\mu} \omega(\mathrm{C}_{\mathsf{p}}m\omega) = \Omega \ .$$

Because there are no other possibilities of reduction the term Ω is not even weakly normalizing. $\square$

We observe that the monotonicity witness does not enter the definition of beta reduction and therefore cannot contribute to normalization in case of trivially monotone fixed-point types (another example being $\mu\alpha.\alpha \to \alpha$ with essentially the same term as above).

Note that with recursive types in type assignment systems it would not be possible to carry monotonicity witnesses around, and since [10] it is known that non-positivity always leads to the typability of non-normalizing terms. The proof above exhibits that the generic non-normalizing term $(\lambda x.xx)(\lambda x.xx)$ of untyped lambda calculus is typable with (monotone) recursive types.

In order to solve the normalization problem in our setting of monotonicity witnesses we define a variant EMF of $\mathsf{EMF}_{\mathsf{naive}}$ by changing the definition of reduction to

$$(\beta_\mu)' \qquad (\mathrm{C}_{\mu\alpha\rho}mt)\mathrm{E}_\mu \mapsto m(\mu\alpha\rho)(\mu\alpha\rho)(\lambda y^{\mu\alpha\rho}y)t \ .$$

Similarly, we define a variant IMF of $\mathsf{IMF}_{\mathsf{naive}}$ by replacing (β_μ) by

$$(\beta_\mu)' \qquad (\mathrm{C}_{\mu\alpha\rho}t)\mathrm{E}_\mu m \mapsto m(\mu\alpha\rho)(\mu\alpha\rho)(\lambda y^{\mu\alpha\rho}y)t \ .$$

[The rules at least make sense in that they do not alter the types and do not introduce new free variables. It may be instructive to compare those rules with the beta rules of iteration on inductive types where the monotonicity witnesses are needed to lift the whole function being defined by iteration such that the recursion argument may be inserted and the result be given as an argument to the step term.]

In EMF and IMF the term in the proof of the above lemma does not pose a problem (we show it for EMF):

$$\Omega \mapsto_{\beta_\to} (\mathrm{C}_{\mathsf{p}}m\omega)\mathrm{E}_\mu(\mathrm{C}_{\mathsf{p}}m\omega) \to_{\beta_\mu{}'} m\mathsf{pp}(\lambda y^{\mathsf{p}}y)\omega(\mathrm{C}_{\mathsf{p}}m\omega) \to^2_{\beta_\forall}\to^3_{\beta_\to} \mathsf{IN1}$$

and there are no other reduction possibilities.

The same difference occurs with the fixed-point combinator for types of the form $\rho \to \rho$ (see [6], p.241) which makes use of the type $\mu\alpha.\alpha \to \rho \to \rho$ with $\alpha \notin \mathsf{FV}(\rho)$ which again has a trivial monotonicity witness.

Theorem 1. *The systems* EMF *and* IMF *are strongly normalizing.*

The proof will be done by embedding the systems into the system NPF of non-interleaved positive fixed-point types which will be defined in the next section.

But let us first comment on the theorem: From Lemma 1 we know that a predecessor for types such as p is incompatible with normalization. The above theorem tells us that strong normalization is gained by changing the beta rule in the described way. Therefore we do not have to impose a positivity restriction but will get a "harmless" term in degenerate cases instead of the predecessor. And the "harmlessness" is simply enforced by the type of m which has to express monotonicity! Section 4 reveals that there are enough non-degenerate cases.

3 Non-Interleaved Positive Fixed-Point Types

The system NPF will have the standard (β_μ)-rule $(\mathrm{C}_{\mu\alpha\rho}t)\mathrm{E}_\mu \mapsto t$ but is strongly normalizing due to a restriction imposed on the formation of types of the form $\mu\alpha\rho$. We only allow $\mu\alpha\rho$ as a type of the system if all free occurrences of α in ρ are at positive positions and not in some subexpression of the form $\mu\beta\sigma$. The formal definition also has to deal with negative positions because we do not restrict ourselves to strict positivity.

Inductively define the set NPTypes of non-interleaved positive types and simultaneously for every $\rho \in \mathsf{NPTypes}$ the set $\mathsf{NPos}(\rho)$ of type variables which only occur free at positive positions in ρ and are not in the scope of an application of the μ-rule and the set $\mathsf{NNeg}(\rho)$ of type variables which only occur free at negative positions in ρ and are not in the scope of an application of the μ-rule as follows:

(V) $\alpha \in \mathsf{NPTypes}$ and $\mathsf{NPos}(\alpha) := \mathsf{Typevars}$ and $\mathsf{NNeg}(\alpha) := \mathsf{Typevars} \setminus \{\alpha\}$.

$(\times)$ If $\rho \in \mathsf{NPTypes}$ and $\sigma \in \mathsf{NPTypes}$, then $\rho \times \sigma \in \mathsf{NPTypes}$.
$\mathsf{NPos}(\rho\times\sigma) := \mathsf{NPos}(\rho)\cap\mathsf{NPos}(\sigma)$ and $\mathsf{NNeg}(\rho\times\sigma) := \mathsf{NNeg}(\rho)\cap\mathsf{Neg}(\sigma)$.

$(\to)$ If $\rho \in \mathsf{NPTypes}$ and $\sigma \in \mathsf{NPTypes}$, then $\rho \to \sigma \in \mathsf{NPTypes}$.
$\mathsf{NPos}(\rho \to \sigma) := \mathsf{NNeg}(\rho) \cap \mathsf{NPos}(\sigma)$.
$\mathsf{NNeg}(\rho \to \sigma) := \mathsf{NPos}(\rho) \cap \mathsf{NNeg}(\sigma)$.

$(\forall)$ If $\rho \in \mathsf{NPTypes}$, then $\forall\alpha\rho \in \mathsf{NPTypes}$.
$\mathsf{NPos}(\forall\alpha\rho) := \mathsf{NPos}(\rho) \cup \{\alpha\}$ and $\mathsf{NNeg}(\forall\alpha\rho) := \mathsf{NNeg}(\rho) \cup \{\alpha\}$.

(μ) If $\rho \in \mathsf{NPTypes}$ and $\alpha \in \mathsf{NPos}(\rho)$, then $\mu\alpha\rho \in \mathsf{NPTypes}$.
$\mathsf{NPos}(\mu\alpha\rho) := \mathsf{NPos}(\rho) \setminus \mathsf{FV}(\mu\alpha\rho)$.
$\mathsf{NNeg}(\mu\alpha\rho) := (\mathsf{NNeg}(\rho) \setminus \mathsf{FV}(\rho)) \cup \{\alpha\}$.

Let NPF be the extension of system F to the types in NPTypes and with the standard rules (μ-I), (μ-E) and (β_μ) mentioned first. (The definitions for system F are still correct because $\mathsf{NPTypes}$ is closed under substitution which is somewhat awkward to prove because one needs additional statements[2] on NPos and NNeg to be proved simultaneously.)

Interleaving μ-types are ruled out in NPF but not nesting! Define $\mathsf{nat} := \mu\alpha.1+\alpha$ (for the definition of 1 see the proof of Lemma 1, $\rho+\sigma := \forall\alpha.(\rho \to \alpha) \to (\sigma \to \alpha) \to \alpha$ with $\alpha \notin \mathsf{FV}(\rho) \cup \mathsf{FV}(\sigma)$) and then define $\mathsf{tree} := \mu\alpha.1 + (\mathsf{nat} \to \alpha)$ which is clearly nested. The following type (due to Ulrich Berger) is not allowed although it is positive:

$$\mathsf{Tree} := \mu\alpha.1 + ((\mu\beta.1 + (\alpha \to \beta)) \to \alpha)$$

NPF is a slight generalization (due to nesting) of F_{ret} in [4] which is shown to be strongly normalizing by an embedding into Mendler's system [10] via a system of non-interleaved positive inductive types with primitive recursion (in [4]).

NPF is a subsystem of the system $\mathsf{M2L}\mu$ in [7] (p. 308) where it is argued (p. 310) that this system may be proved strongly normalizing by adopting the same method as for Mendler's system.

In [9] a direct proof is given of strong normalization of the variant of Mendler's system which simply leaves out the positivity requirement in [10] (which is crucial to the normalization proof presented there). This more liberal system also embeds in a reduction-preserving way into NPF which will be sketched in the final section. Therefore the author thinks that a direct proof of strong normalization for NPF gives an easier access to the above-mentioned normalization results.

Before turning to normalization proofs we try to see in which way NPF embeds into the systems of monotone fixed-point types.

4 Embedding NPF into Monotone Fixed-Point Types

Clearly, we expect non-interleaved positive fixed-point types to be monotone fixed-point types. This will be studied more carefully in this section. It turns out that in presence of eta rules we indeed get an embedding.

4.1 Definition of Monotonicity Witnesses for NPF

We want to define closed terms $\mathsf{map}_{\lambda\alpha\rho}$ of type $\forall\alpha\forall\beta.(\alpha \to \beta) \to \rho \to \rho[\alpha := \beta]$ in NPF for every $\mu\alpha\rho \in \mathsf{NPTypes}$, i.e. $\rho \in \mathsf{NPTypes}$ and $\alpha \in \mathsf{NPos}(\rho)$. Because positivity and negativity are defined simultaneously, we have to define auxiliary terms $\mathsf{comap}_{\lambda\alpha\rho}$ for $\alpha \in \mathsf{NNeg}(\rho)$ simultaneously (which will show up in the case of arrow types):

Define for every type $\rho \in \mathsf{NPTypes}$ and every $\alpha \in \mathsf{NPos}(\rho)$ (and $\beta \notin \{\alpha\} \cup \mathsf{FV}(\rho)$) a closed term

$$\mathsf{map}_{\lambda\alpha\rho} : \forall\alpha\forall\beta.(\alpha \to \beta) \to \rho \to \rho[\alpha := \beta]$$

[2] One has to prove: Let ρ, $\boldsymbol{\sigma} \in \mathsf{NPTypes}$. Then $\rho[\boldsymbol{\alpha} := \boldsymbol{\sigma}] \in \mathsf{NPTypes}$ and $\mathsf{NPos}(\rho) \setminus \mathsf{FV}(\boldsymbol{\sigma}) \subseteq \mathsf{NPos}(\rho[\boldsymbol{\alpha} := \boldsymbol{\sigma}])$ and $\mathsf{NNeg}(\rho) \setminus \mathsf{FV}(\boldsymbol{\sigma}) \subseteq \mathsf{NNeg}(\rho[\boldsymbol{\alpha} := \boldsymbol{\sigma}])$. Moreover, if $\beta \notin \mathsf{FV}(\rho)$ then $\alpha \in \mathsf{NPos}(\rho)$ implies $\beta \in \mathsf{NPos}(\rho[\alpha := \beta])$ and $\alpha \in \mathsf{NNeg}(\rho)$ implies $\beta \in \mathsf{NNeg}(\rho[\alpha := \beta])$.

and for every $\alpha \in \mathsf{NNeg}(\rho)$ (and $\beta \notin \{\alpha\} \cup \mathsf{FV}(\rho)$) a closed term

$$\mathsf{comap}_{\lambda\alpha\rho} : \forall\alpha\forall\beta.(\alpha \to \beta) \to \rho[\alpha := \beta] \to \rho$$

by recursion on ρ.

A definition which yields normal terms is shown in [9] pp. 73–75. Because we do not allow for interleaving one only has to omit the case (μ). The reader also finds a definition in [7] p. 311. Again one has to omit the μ-clause and should make a case distinction whether α occurs free in ρ or not. Another place is [4] p. 202 which essentially contains the definitions. Because we have no interleaving it does not matter that those definitions are all made for systems of *inductive* types (i. e. $\mu\alpha\rho$ denotes a least fixed-point), and moreover, the definition is quite straightforward and by structural recursion on ρ.

4.2 The Embeddings

An embedding $-'$ of NPF in $\mathsf{EMF}_{\mathsf{naive}}$ is now immediately defined by induction on the terms of NPF setting

$$(\mu\text{-I}) \quad (\mathrm{C}_{\mu\alpha\rho} t)' := \mathrm{C}_{\mu\alpha\rho} \mathsf{map}_{\lambda\alpha\rho} t'.$$

and homomorphic rules for all the other term formers. This gives terms in $\mathsf{EMF}_{\mathsf{naive}}$ because $\mathsf{map}_{\lambda\alpha\rho}$ is built without $(\mu\text{-I})$ and hence already is a term of $\mathsf{EMF}_{\mathsf{naive}}$.

It is easy to see that if $r \to s$ in NPF, then $r' \to s'$ in $\mathsf{EMF}_{\mathsf{naive}}$. (For (β_μ) we use that $\mathsf{map}_{\lambda\alpha\rho}$ is a closed term, for $(\beta_\forall)$ we have to prove that type substitution commutes with forming $\mathsf{map}_{\lambda\alpha\rho}$.)

It is clear how to change the embedding to an embedding of NPF into $\mathsf{IMF}_{\mathsf{naive}}$.

Unfortunately, the two embeddings do not preserve (β_μ) in EMF and IMF, respectively. An easy solution to this problem is the addition of η-rules for $\times$, $\to$ and $\forall$ as usual:

$$\begin{array}{cll}
(\eta_\times) & \langle r\mathsf{L}, r\mathsf{R}\rangle \mapsto r & \\
(\eta_\to) & \lambda x^\rho.rx \mapsto r & \text{if } x \notin \mathsf{FV}(r) \\
(\eta_\forall) & \Lambda\alpha.r\alpha \mapsto r & \text{if } \alpha \notin \mathsf{FTV}(r)
\end{array}$$

(where $\mathsf{FTV}(r)$ is the set of free type variables of r).

Indicate the presence of these additional rules by the index eta in the names of the systems.

Now it is easy to see that in $\mathsf{NPF}_{\mathsf{eta}}$ (and trivially also in $\mathsf{EMF}_{\mathsf{eta}}$ and $\mathsf{IMF}_{\mathsf{eta}}$) we have that map and comap satisfy the first functor law, more precisely

$$\mathsf{map}_{\lambda\alpha\rho}\sigma\sigma(\lambda x^\sigma x)t^{\rho[\alpha:=\sigma]} \to^* t \text{ and } \mathsf{comap}_{\lambda\alpha\rho}\sigma\sigma(\lambda x^\sigma x)t^{\rho[\alpha:=\sigma]} \to^* t$$

(where $\to^*$ denotes the reflexive and transitive closure of $\to$).

Therefore we get that whenever $r \to s$ in NPF, then $r' \to^+ s'$ in the systems $\mathsf{EMF}_{\mathsf{eta}}$ and $\mathsf{IMF}_{\mathsf{eta}}$, respectively (where $\to^+$ denotes the transitive closure of $\to$).

This entails that from strong normalization of $\mathsf{EMF}_{\mathsf{eta}}$ or $\mathsf{IMF}_{\mathsf{eta}}$, we can infer strong normalization of NPF.

But now we show directly that even $\mathsf{NPF}_{\mathsf{eta}}$ is strongly normalizing and derive strong normalization of $\mathsf{EMF}_{\mathsf{eta}}$ and $\mathsf{IMF}_{\mathsf{eta}}$ from this fact via an embedding.

We show even more:

Let $\mathsf{NPF}_{\mathsf{eta}+}$ be the system $\mathsf{NPF}_{\mathsf{eta}}$ extended by the η-rule for fixed-point types:

$$(\eta_\mu) \qquad \mathrm{C}_{\mu\alpha\rho}(r^{\mu\alpha\rho}\mathrm{E}_\mu) \mapsto r$$

This rule justifies the name "fixed-point types" instead of retract types.

Theorem 2. *The system* $\mathsf{NPF}_{\mathsf{eta}+}$ *is strongly normalizing.*

5 A Direct Proof of Strong Normalization for $\mathsf{NPF}_{\mathsf{eta}+}$

Every term and type in this section is taken from $\mathsf{NPF}_{\mathsf{eta}+}$.

5.1 Head Forms, the Set SN and Saturated Sets

Lemma 2 (On head forms). *Every term has exactly one of the following forms:*

- (V) $x^\rho \boldsymbol{s}$
- ($\times$-I) $\langle r, s\rangle$
- ($\times$-R) $\langle r, s\rangle \mathsf{L}\boldsymbol{s}$, $\langle r, s\rangle \mathsf{R}\boldsymbol{s}$
- ($\to$-I) $\lambda x^\rho r$
- ($\to$-R) $(\lambda x^\rho r)s\boldsymbol{s}$
- ($\forall$-I) $\Lambda\alpha r$
- ($\forall$-R) $(\Lambda\alpha r)\sigma\boldsymbol{s}$
- (μ-I) $\mathrm{C}_{\mu\alpha\rho}t$
- (μ-R) $(\mathrm{C}_{\mu\alpha\rho}t)\mathrm{E}_\mu\boldsymbol{s}$

The vector $\boldsymbol{s}$ denotes a list made up of the symbols L, R and E_μ and of terms and types such that e. g. $x^\rho\boldsymbol{s}$ is a well-typed term. Hence, the vector notation is a means of expressing multiple eliminations.

Proof. Very easy induction on terms. □

Define the set SN of terms by (strictly positive) induction as follows:

- (V) If $x^\rho\boldsymbol{s}$ is a term and the terms in $\boldsymbol{s}$ are in SN, then $x^\rho\boldsymbol{s} \in \mathsf{SN}$.
- ($\times$-I) If $r \in \mathsf{SN}$ and $s \in \mathsf{SN}$, then $\langle r, s\rangle \in \mathsf{SN}$.
- ($\beta_\times$) If $r\boldsymbol{s} \in \mathsf{SN}$ and $s \in \mathsf{SN}$, then $\langle r, s\rangle\mathsf{L}\boldsymbol{s} \in \mathsf{SN}$.
  If $s\boldsymbol{s} \in \mathsf{SN}$ and $r \in \mathsf{SN}$, then $\langle r, s\rangle\mathsf{R}\boldsymbol{s} \in \mathsf{SN}$.
- ($\to$-I) If $r \in \mathsf{SN}$, then $\lambda x^\rho r \in \mathsf{SN}$.
- ($\beta_\to$) If $(\lambda x^\rho r)s\boldsymbol{s}$ is a term, $s \in \mathsf{SN}$ and $r[x^\rho := s]\boldsymbol{s} \in \mathsf{SN}$, then $(\lambda x^\rho r)s\boldsymbol{s} \in \mathsf{SN}$.
- ($\forall$-I) If $\Lambda\alpha r$ is a term and $r \in \mathsf{SN}$, then $\Lambda\alpha r \in \mathsf{SN}$.

($\beta_\forall$) If $(\Lambda\alpha r)\sigma\boldsymbol{s}$ is a term and $r[\alpha := \sigma]\boldsymbol{s} \in \mathsf{SN}$, then $(\Lambda\alpha r)\sigma\boldsymbol{s} \in \mathsf{SN}$.
(μ-I) If $t \in \mathsf{SN}$, then $\mathrm{C}_{\mu\alpha\rho}t \in \mathsf{SN}$.
(μ-R) If $(\mathrm{C}_{\mu\alpha\rho}t)\mathrm{E}_\mu\boldsymbol{s}$ is a term and $t\boldsymbol{s} \in \mathsf{SN}$, then $(\mathrm{C}_{\mu\alpha\rho}t)\mathrm{E}_\mu\boldsymbol{s} \in \mathsf{SN}$.

Lemma 3. SN *is the set* sn *of strongly normalizing terms.*

Proof. $\mathsf{sn} \subseteq \mathsf{SN}$ is shown by main induction on the height of the reduction tree and side induction on the term height. (This part of the lemma is not needed for the proof of strong normalization.) $\mathsf{SN} \subseteq \mathsf{sn}$ is proved by showing that the defining clauses of SN are closure properties of the set sn. To my knowledge it is unavoidable to check those closure properties if one wants to prove strong normalization. See e. g. [9] for an exposition. $\square$

The idea of using such a characterization of the set of strongly normalizing terms (at least for the untyped λ-calculus) was brought to my attention by [11] and [8].

I want to stress that it is invisible in this approach that we deal with $\mathsf{NPF}_{\mathsf{eta}+}$ instead of NPF—at least it does not enter the definitions. Only when showing $\mathsf{SN} \subseteq \mathsf{sn}$ one has to consider additional cases which are trivial to handle.[3]

Let SN_τ be the set of terms of type τ in SN. A set $\mathcal{M}$ of terms is called τ-saturated, if the following conditions are met.

(SN) If $r \in \mathcal{M}$, then $r \in \mathsf{SN}_\tau$.
(V) If $x^\rho\boldsymbol{s}$ is a term of type τ and the terms in $\boldsymbol{s}$ are in SN, then $x^\rho\boldsymbol{s} \in \mathcal{M}$.
($\beta_\times$) If $r\boldsymbol{s} \in \mathcal{M}$ and $s \in \mathsf{SN}$, then $\langle r, s\rangle\mathsf{L}\boldsymbol{s} \in \mathcal{M}$. If $s\boldsymbol{s} \in \mathcal{M}$ and $r \in \mathsf{SN}$, then $\langle r, s\rangle\mathsf{R}\boldsymbol{s} \in \mathcal{M}$.
($\beta_\to$) If $(\lambda x^\rho r)s\boldsymbol{s}$ is a term, $s \in \mathsf{SN}$ and $r[x^\rho := s]\boldsymbol{s} \in \mathcal{M}$, then $(\lambda x^\rho r)s\boldsymbol{s} \in \mathcal{M}$.
($\beta_\forall$) If $(\Lambda\alpha r)\sigma\boldsymbol{s}$ is a term and $r[\alpha := \sigma]\boldsymbol{s} \in \mathcal{M}$, then $(\Lambda\alpha r)\sigma\boldsymbol{s} \in \mathcal{M}$.
(β_μ) If $(\mathrm{C}_{\mu\alpha\rho}t)\mathrm{E}_\mu\boldsymbol{s}$ is a term and $t\boldsymbol{s} \in \mathcal{M}$, then $(\mathrm{C}_{\mu\alpha\rho}t)\mathrm{E}_\mu\boldsymbol{s} \in \mathcal{M}$.

Let SAT_τ be the set of all τ-saturated sets. Obviously, $\mathsf{SN}_\tau \in \mathsf{SAT}_\tau$.

Let M be a set of terms. The τ-saturated closure $\mathsf{cl}_\tau(M)$ of M is defined to be the smallest τ-saturated set containing $M \cap \mathsf{SN}_\tau$. (This may be defined by a strictly positive inductive definition.)

The candidate method goes as follows: By means of saturated sets we define (by recursion on types) predicates of strong computability with respect to an assignment of saturated sets for type variables (a candidate assignment) and finally show (by induction on terms) that every term is strongly computable under substitution. Hence every term is contained in a saturated set (the computability predicate) which only consists of strongly normalizing terms (due to the preceding lemma).

This all works if there are constructions for saturated sets corresponding to the type constructs of the system for which the introduction rules and the elimination rules are sound. (This is presented at length in [9].)

[3] The reason is: All of the systems are weakly orthogonal which amounts to checking that the critical pairs between beta and eta rules are trivial.

5.2 Calculating with Saturated Sets

Here we concentrate on the constructions of saturated sets corresponding to the type constructs.

Let $\mathcal{M} \in \mathsf{SAT}_\rho$ and $\mathcal{N} \in \mathsf{SAT}_\sigma$. Define

$$\begin{aligned} M_I &:= \{r^{\rho\times\sigma} | \exists s \in \mathcal{M} \exists t \in \mathcal{N} . r = \langle s, t\rangle\} \\ M_E &:= \{r^{\rho\times\sigma} | r\mathsf{L} \in \mathcal{M} \wedge r\mathsf{R} \in \mathcal{N}\} \end{aligned}$$

and set $\mathcal{M} \times_I \mathcal{N} := \mathsf{cl}_{\rho\times\sigma}(M_I)$ and $\mathcal{M} \times_E \mathcal{N} := \mathsf{cl}_{\rho\times\sigma}(M_E)$. Hence, $\mathcal{M} \times_I \mathcal{N}$ and $\mathcal{M} \times_E \mathcal{N}$ are $(\rho \times \sigma)$-saturated sets and the construction is isotone in the arguments $\mathcal{M}$ and $\mathcal{N}$.

Writing $\mathcal{M} \times \mathcal{N}$ for $\mathcal{M} \times_I \mathcal{N}$ or $\mathcal{M} \times_E \mathcal{N}$ (for both of them) we get (after some reasoning)

($\times$-I) If $r \in \mathcal{M}$ and $s \in \mathcal{N}$, then $\langle r, s\rangle \in \mathcal{M} \times \mathcal{N}$.
($\times$-E) If $r \in \mathcal{M} \times \mathcal{N}$, then $r\mathsf{L} \in \mathcal{M}$ and $r\mathsf{R} \in \mathcal{N}$.

Let $\mathcal{M} \in \mathsf{SAT}_\rho$ and $\mathcal{N} \in \mathsf{SAT}_\sigma$. Define

$$\begin{aligned} M_I &:= \{r^{\rho\to\sigma} | \exists x^\rho \exists t^\sigma . r = \lambda x^\rho t \wedge \forall s \in \mathcal{M}\, t[x^\rho := s] \in \mathcal{N}\} \\ M_E &:= \{r^{\rho\to\sigma} | \forall s \in \mathcal{M}\, rs \in \mathcal{N}\} \end{aligned}$$

and set $\mathcal{M} \to_I \mathcal{N} := \mathsf{cl}_{\rho\to\sigma}(M_I)$ and $\mathcal{M} \to_E \mathcal{N} := \mathsf{cl}_{\rho\to\sigma}(M_E)$. Hence, $\mathcal{M} \to_I \mathcal{N}$ and $\mathcal{M} \to_E \mathcal{N}$ are $(\rho \to \sigma)$-saturated sets and the construction is isotone in the argument $\mathcal{N}$ and antitone in the argument $\mathcal{M}$.

Writing $\mathcal{M} \to \mathcal{N}$ for $\mathcal{M} \to_I \mathcal{N}$ or $\mathcal{M} \to_E \mathcal{N}$ (for both of them) we get (after some reasoning)

($\to$-I) If $t[x^\rho := s] \in \mathcal{N}$ for all $s \in \mathcal{M}$, then $\lambda x^\rho t \in \mathcal{M} \to \mathcal{N}$.
($\to$-E) If $r \in \mathcal{M} \to \mathcal{N}$ and $s \in \mathcal{M}$, then $rs \in \mathcal{N}$.

Fix $\lambda\alpha\rho$. Let $\Phi := (\Phi_\tau)_\tau$ be a family of mappings $\Phi_\tau : \mathsf{SAT}_\tau \to \mathsf{SAT}_{\rho[\alpha:=\tau]}$. Define

$$\begin{aligned} M_I &:= \{r^{\forall\alpha\rho} | \exists\alpha\exists t^\rho . r = \Lambda\alpha t \wedge \forall\tau\forall\mathcal{P} \in \mathsf{SAT}_\tau\, t[\alpha := \tau] \in \Phi_\tau(\mathcal{P})\} \\ M_E &:= \{r^{\forall\alpha\rho} | \forall\tau\forall\mathcal{P} \in \mathsf{SAT}_\tau\, r\tau \in \Phi_\tau(\mathcal{P})\} \end{aligned}$$

and set $\forall_I(\Phi) := \mathsf{cl}_{\forall\alpha\rho}(M_I)$ and $\forall_E(\Phi) := \mathsf{cl}_{\forall\alpha\rho}(M_E)$. Hence, $\forall_I(\Phi)$ and $\forall_E(\Phi)$ are $\forall\alpha\rho$-saturated sets and the construction is (pointwise) isotone in the argument Φ.

Writing $\forall(\Phi)$ for $\forall_I(\Phi)$ or $\forall_E(\Phi)$ (for both of them) we get (after some reasoning)

($\forall$-I) If $\forall\tau\forall\mathcal{P} \in \mathsf{SAT}_\tau\, t[\alpha := \tau] \in \Phi_\tau(\mathcal{P})$, then $\Lambda\alpha t \in \forall(\Phi)$.
($\forall$-E) If $r \in \forall(\Phi)$, then $\forall\tau\forall\mathcal{P} \in \mathsf{SAT}_\tau\, r\tau \in \Phi_\tau(\mathcal{P})$.

We now come to the interesting case of fixed-point types. Fix $\lambda\alpha\rho$. Let ϕ be an isotone mapping from $\mathsf{SAT}_{\mu\alpha\rho}$ to $\mathsf{SAT}_{\rho[\alpha:=\mu\alpha\rho]}$. Define for $\mathcal{M} \in \mathsf{SAT}_{\mu\alpha\rho}$

$$\begin{aligned} M_I(\mathcal{M}) &:= \{r^{\mu\alpha\rho} | \exists t \in \phi(\mathcal{M}).r = \mathrm{C}_{\mu\alpha\rho}t\} \\ M_E(\mathcal{M}) &:= \{r^{\mu\alpha\rho} | r\mathrm{E}_\mu \in \phi(\mathcal{M})\} \end{aligned}$$

and $\Psi_I : \mathsf{SAT}_{\mu\alpha\rho} \to \mathsf{SAT}_{\mu\alpha\rho}$ and $\Psi_E : \mathsf{SAT}_{\mu\alpha\rho} \to \mathsf{SAT}_{\mu\alpha\rho}$ by setting $\Psi_I(\mathcal{M}) := \mathsf{cl}_{\mu\alpha\rho}(M_I(\mathcal{M}))$ and $\Psi_E(\mathcal{M}) := \mathsf{cl}_{\mu\alpha\rho}(M_E(\mathcal{M}))$ and set

$$\mu_I(\phi) := \text{ any fixed point of } \Psi_I \quad \text{and} \quad \mu_E(\phi) := \text{ any fixed point of } \Psi_E.$$

If there are several fixed-points, then μ_I and μ_E need not be isotone in ϕ. Of course, there is always a fixed-point because Ψ_I and Ψ_E are monotone endofunctions on the complete lattice $\mathsf{SAT}_{\mu\alpha\rho}$. This time the work behind "after some reasoning" is spelled out precisely. It is the suite of lemmas which may nearly be written down mechanically.

Lemma 4. *$M_I(\mathcal{M}) \subseteq \mathsf{SN}$ and $M_E(\mathcal{M}) \cap \mathsf{SN} \in \mathsf{SAT}_{\mu\alpha\rho}$.*

Proof. First part obvious. Second part: One only has to append E_μ to $\boldsymbol{s}$. □

Lemma 5. *$\mathcal{M} \in \mathsf{SAT}_{\mu\alpha\rho}$ is a pre-fixed point of Ψ_I iff $\forall t \in \phi(\mathcal{M}).C_{\mu\alpha\rho}t \in \mathcal{M}$. $\mathcal{M}$ is a post-fixed point of Ψ_E iff $\forall r \in \mathcal{M}.rE_\mu \in \phi(\mathcal{M})$.*

Proof. The preceding lemma yields $M_I(\mathcal{M}) \subseteq \Psi_I(\mathcal{M})$ and $\Psi_E(\mathcal{M}) \subseteq M_E(\mathcal{M})$. Moreover, for $\mathcal{N} \in \mathsf{SAT}_{\mu\alpha\rho}$ we have that $M_I(\mathcal{M}) \subseteq \mathcal{N}$ implies $\Psi_I(\mathcal{M}) \subseteq \mathcal{N}$ and $\mathcal{N} \subseteq M_E(\mathcal{M})$ implies $\mathcal{N} \subseteq \Psi_E(\mathcal{M})$. □

Lemma 6. *$M_I(\mathcal{M}) \subseteq M_E(\mathcal{M})$ for every $\mathcal{M} \in \mathsf{SAT}_{\mu\alpha\rho}$.*

Proof. Let $t \in \phi(\mathcal{M})$. We have to show that $\mathrm{C}_{\mu\alpha\rho}t \in M_E(\mathcal{M})$. Hence, we have to show that $(\mathrm{C}_{\mu\alpha\rho}t)\mathrm{E}_\mu \in \phi(\mathcal{M})$. Because $\phi(\mathcal{M})$ is saturated it suffices to show $t \in \phi(\mathcal{M})$. (Note that this is the only explicit reference to the clause (β_μ) in the definition of saturated sets.) □

Lemma 7. *$\mu_E(\phi)$ is a pre-fixed point of Ψ_I and $\mu_I(\phi)$ is a post-fixed point of Ψ_E.*

Proof. Show that $\Psi_I(\mu_E(\phi)) \subseteq \mu_E(\phi)$: Because $\mu_E(\phi)$ is a pre-fixed point of Ψ_E it suffices to show that $\Psi_I(\mu_E(\phi)) \subseteq \Psi_E(\mu_E(\phi))$. Because $\mathsf{cl}_{\mu\alpha\rho}$ is monotone, it suffices to show $M_I(\mu_E(\phi)) \subseteq M_E(\mu_E(\phi))$ which is an instance of the preceding lemma. $\mu_I(\phi) \subseteq \Psi_E(\mu_I(\phi))$ is established similarly. □

Writing $\mu(\phi)$ for $\mu_I(\phi)$ or $\mu_E(\phi)$ (for both of them) we finally get

(μ-I) $\forall t \in \phi(\mu(\phi)).\mathrm{C}_{\mu\alpha\rho}t \in \mu(\phi)$.
(μ-E) $\forall r \in \mu(\phi).r\mathrm{E}_\mu \in \phi(\mu(\phi))$.

5.3 Computability Predicates

Now the saturated sets are used to define the computability predicates which may be seen as a semantics of the types.

Define the ($\rho[\boldsymbol{\alpha} := \boldsymbol{\sigma}]$)-saturated set $\mathsf{SC}_\rho[\boldsymbol{\alpha} := \boldsymbol{\mathcal{P}}]$ of strongly computable terms w.r.t. the type ρ, the finite list of type variables $\boldsymbol{\alpha}$ and the (equally long) list of saturated sets $\boldsymbol{\mathcal{P}}$, where σ_i is the unique type such that $\mathcal{P}_i$ is σ_i-saturated, by recursion on ρ and simultaneously prove that $\mathsf{SC}_\rho[\boldsymbol{\alpha} := \boldsymbol{\mathcal{P}}]$ is an isotone function of $\mathcal{P}_i$ for $\alpha_i \in \mathsf{NPos}(\rho)$ and an antitone function of $\mathcal{P}_i$ for $\alpha_i \in \mathsf{NNeg}(\rho)$ and does not depend on $\mathcal{P}_i$ if $\alpha \notin \mathsf{FV}(\rho)$ as follows.

(V) $\mathsf{SC}_\alpha[\boldsymbol{\alpha} := \boldsymbol{\mathcal{P}}] := \mathsf{cl}_\alpha(\Lambda)$, if α does not occur in the list $\boldsymbol{\alpha}$. (Note that $\mathsf{cl}_\alpha(\Lambda) = \mathsf{SN}_\alpha$.)
$\mathsf{SC}_\alpha[\boldsymbol{\alpha} := \boldsymbol{\mathcal{P}}] := \mathcal{P}_i$, if i is the smallest index such that $\alpha_i = \alpha$.
($\times$) $\mathsf{SC}_{\rho\times\sigma}[\boldsymbol{\alpha} := \boldsymbol{\mathcal{P}}] := \mathsf{SC}_\rho[\boldsymbol{\alpha} := \boldsymbol{\mathcal{P}}] \times \mathsf{SC}_\sigma[\boldsymbol{\alpha} := \boldsymbol{\mathcal{P}}]$.
($\to$) $\mathsf{SC}_{\rho\to\sigma}[\boldsymbol{\alpha} := \boldsymbol{\mathcal{P}}] := \mathsf{SC}_\rho[\boldsymbol{\alpha} := \boldsymbol{\mathcal{P}}] \to \mathsf{SC}_\sigma[\boldsymbol{\alpha} := \boldsymbol{\mathcal{P}}]$.
($\forall$) $\mathsf{SC}_{\forall\alpha\rho}[\boldsymbol{\alpha} := \boldsymbol{\mathcal{P}}] := \forall((\Phi_\sigma)_\sigma)$, where $\Phi_\sigma : \mathsf{SAT}_\sigma \to \mathsf{SAT}_{\rho[\boldsymbol{\alpha},\alpha:=\boldsymbol{\sigma},\sigma]}$ is defined by $\Phi_\sigma(\mathcal{P}) := \mathsf{SC}_\rho[\boldsymbol{\alpha},\alpha := \boldsymbol{\mathcal{P}},\mathcal{P}]$ (We assume that $\alpha \notin \boldsymbol{\alpha} \cup \mathsf{FV}(\boldsymbol{\sigma})$).
(μ) $\mathsf{SC}_{\mu\alpha\rho}[\boldsymbol{\alpha} := \boldsymbol{\mathcal{P}}] := \mu(\phi)$, where $\phi : \mathsf{SAT}_{\mu\alpha\rho} \to \mathsf{SAT}_{\rho[\boldsymbol{\alpha},\alpha:=\boldsymbol{\sigma},\mu\alpha\rho]}$ is defined by $\phi(\mathcal{P}) := \mathsf{SC}_\rho[\boldsymbol{\alpha},\alpha := \boldsymbol{\mathcal{P}},\mathcal{P}]$ (with the same assumption as for ($\forall$)). In this case it is important to have the simultaneously proved statements at hand. Thanks to the restriction to non-interleaved fixed-point types they can be derived from the induction hypothesis.

It is now standard technique first to prove a substitution lemma for computability predicates (by induction on types) and second to prove that every term is strongly computable under substitution (by induction on terms) which gives the theorem to be proved. (For system F and extensions thereof by inductive types this is done in all details in [9].)

6 Embedding EMF and IMF into NPF

Define for every type ρ of EMF a type ρ' of NPF with the only non-homomorphic rule:

$$(\mu\alpha\rho)' := \mu\alpha\forall\beta.(\alpha \to \beta) \to \rho'[\alpha := \beta] \ .$$

It is easy to see that we indeed get non-interleaving (non-strictly) positive fixed-point types and that ρ and ρ' have the same free type variables.

Lemma 8. $(\rho[\boldsymbol{\alpha} := \boldsymbol{\sigma}])' = \rho'[\boldsymbol{\alpha} := \boldsymbol{\sigma}']$.

Proof. Induction on ρ. □

Define for every term r^ρ of EMF a term r' of type ρ' of NPF as follows (the other clauses are homomorphic):

(μ-I) $(\mathrm{C}_{\mu\alpha\rho} m t)' := \mathrm{C}_{(\mu\alpha\rho)'}\left(\Lambda\beta\lambda y^{(\mu\alpha\rho)'\to\beta}.m'(\mu\alpha\rho)'\beta y t'\right)$

(μ-E) $(r^{\mu\alpha\rho}\mathrm{E}_\mu)' := r'\mathrm{E}_\mu(\mu\alpha\rho)'(\lambda x^{(\mu\alpha\rho)'}x)$.

[More precisely, we simultaneously have to prove that r' has the same type variables as r and that x^τ is a free variable of r' iff $\tau = \sigma'$ and x^σ is free in r.]

Lemma 9. *$(r[\boldsymbol{x}^{\boldsymbol{\rho}} := \boldsymbol{s}^{\boldsymbol{\rho}}])' = r'[\boldsymbol{x}^{\boldsymbol{\rho}'} := \boldsymbol{s}']$ and $(r[\boldsymbol{\alpha} := \boldsymbol{\sigma}])' = r'[\boldsymbol{\alpha} := \boldsymbol{\sigma}']$.*

Proof. Induction on r. □

Lemma 10. *If $r \to \hat{r}$, then $r' \to^+ \hat{r}'$.*

Proof. Induction on $r \to \hat{r}$: The cases $(\beta_\to)$ and $(\beta_\forall)$ follow immediately from the preceding lemma. $(\beta_\times)$ is trivial. (β_μ):

$$\begin{aligned}
((\mathrm{C}_{\mu\alpha\rho}mt)\mathrm{E}_\mu)' &= \mathrm{C}_{(\mu\alpha\rho)'}\Big(\Lambda\beta\lambda y^{(\mu\alpha\rho)'\to\beta}.m'(\mu\alpha\rho)'\beta y t'\Big)\mathrm{E}_\mu(\mu\alpha\rho)'(\lambda x^{(\mu\alpha\rho)'}x) \\
&\to \Big(\Lambda\beta\lambda y^{(\mu\alpha\rho)'\to\beta}.m'(\mu\alpha\rho)'\beta y t'\Big)(\mu\alpha\rho)'(\lambda x^{(\mu\alpha\rho)'}x) \\
&\to \Big(\lambda y^{(\mu\alpha\rho)'\to(\mu\alpha\rho)'}.m'(\mu\alpha\rho)'(\mu\alpha\rho)' y t'\Big)(\lambda x^{(\mu\alpha\rho)'}x) \\
&\mapsto m'(\mu\alpha\rho)'(\mu\alpha\rho)'(\lambda x^{(\mu\alpha\rho)'}x)t' \\
&= \Big(m(\mu\alpha\rho)(\mu\alpha\rho)(\lambda x^{\mu\alpha\rho}x)t\Big)'
\end{aligned}$$

The term closure clauses are easily proved by using the induction hypothesis. □

In order to see how the embedding of IMF works it is convenient to introduce existential types into NPF which can later be eliminated in a reduction-preserving way.[4] Hence, we embed IMF into $\mathsf{NPF}_{\mathsf{ex}}$ which is NPF extended by types of the form $\exists\alpha\rho$ (more formally we have to extend the definition of $\mathsf{NPTypes}$), by the term rules

($\exists$-I) If $t^{\rho[\alpha:=\tau]}$ is a term, then $(\mathrm{C}_{\exists\alpha\rho,\tau}t)^{\exists\alpha\rho}$ is a term.
($\exists$-E) If $r^{\exists\alpha\rho}$ is a term, $s^{\forall\alpha.\rho\to\sigma}$ is a term and $\alpha \notin \mathsf{FV}(\sigma)$, then $(r\mathrm{E}_\exists s)^\sigma$ is a term.

and by the reduction rule

$$(\beta_\exists) \qquad (C_{\exists\alpha\rho,\tau}t)\mathrm{E}_\exists s \mapsto s\tau t \ .$$

[These are clearly the rules pertaining to the second-order existential quantifier in systems of natural deduction together with a sound transformation rule.]

Define for every type ρ of IMF a type ρ' of $\mathsf{NPF}_{\mathsf{ex}}$ with the only non-trivial rule:

$$(\mu\alpha\rho)' := \mu\alpha\exists\beta.(\beta \to \alpha) \times \rho'[\alpha := \beta] \ .$$

We even get strictly-positive non-interleaving fixed-point types (if strict positivity is defined in a way which allows existential quantifiers).

Lemma 8 also holds for this definition.

Define for every term r^ρ of IMF a term r' of type ρ' of $\mathsf{NPF}_{\mathsf{ex}}$ as follows (the other clauses are homomorphic):

[4] The well-known impredicative coding to be found e.g. in [5] p. 86 also preserves reduction.

$$(\mu\text{-I})\quad (\mathrm{C}_{\mu\alpha\rho}t)' := \mathrm{C}_{(\mu\alpha\rho)'}\mathrm{C}_{\exists\beta.(\beta\to(\mu\alpha\rho)')\times\rho'[\alpha:=\beta],(\mu\alpha\rho)'}\langle\lambda x^{(\mu\alpha\rho)'}x, t'\rangle$$

$$(\mu\text{-E})\quad (r^{\mu\alpha\rho}\mathrm{E}_\mu m)' := r'\mathrm{E}_\mu\mathrm{E}_\exists\Big(\Lambda\beta\lambda z^{(\beta\to(\mu\alpha\rho)')\times\rho'[\alpha:=\beta]}.m'\beta(\mu\alpha\rho)'(z\mathsf{L})(z\mathsf{R})\Big)\ .$$

Lemma 9 also holds for this definition.

The proof that we indeed get an embedding is changed only for (β_μ): Set $s := \Lambda\beta\lambda z^{(\beta\to(\mu\alpha\rho)')\times\rho'[\alpha:=\beta]}.m'\beta(\mu\alpha\rho)'(z\mathsf{L})(z\mathsf{R})$.

$$\begin{aligned}
((\mathrm{C}_{\mu\alpha\rho}t)\mathrm{E}_\mu m)' &= \mathrm{C}_{(\mu\alpha\rho)'}\mathrm{C}_{\exists\beta.(\beta\to(\mu\alpha\rho)')\times\rho'[\alpha:=\beta],(\mu\alpha\rho)'}\langle\lambda x^{(\mu\alpha\rho)'}x, t'\rangle\mathrm{E}_\mu\mathrm{E}_\exists s\\
&\to \mathrm{C}_{\exists\beta.(\beta\to(\mu\alpha\rho)')\times\rho'[\alpha:=\beta],(\mu\alpha\rho)'}\langle\lambda x^{(\mu\alpha\rho)'}x, t'\rangle\mathrm{E}_\exists s\\
&\mapsto s(\mu\alpha\rho)'\langle\lambda x^{(\mu\alpha\rho)'}x, t'\rangle\\
&\to \Big(\lambda z^{\cdots}.m'(\mu\alpha\rho)'(\mu\alpha\rho)'(z\mathsf{L})(z\mathsf{R})\Big)\langle\lambda x^{(\mu\alpha\rho)'}x, t'\rangle\\
&\mapsto m'(\mu\alpha\rho)'(\mu\alpha\rho)'(\langle\lambda x^{(\mu\alpha\rho)'}x, t'\rangle\mathsf{L})(\langle\lambda x^{(\mu\alpha\rho)'}x, t'\rangle\mathsf{R})\\
&\to m'(\mu\alpha\rho)'(\mu\alpha\rho)'(\lambda x^{(\mu\alpha\rho)'}x)(\langle\lambda x^{(\mu\alpha\rho)'}x, t'\rangle\mathsf{R})\\
&\to m'(\mu\alpha\rho)'(\mu\alpha\rho)'(\lambda x^{(\mu\alpha\rho)'}x)t'\\
&= \Big(m(\mu\alpha\rho)(\mu\alpha\rho)(\lambda x^{\mu\alpha\rho}x)t\Big)'
\end{aligned}$$

It is clear that the embeddings are also embeddings of $\mathsf{EMF}_{\mathsf{eta}}$ and $\mathsf{IMF}_{\mathsf{eta}}$ into $\mathsf{NPF}_{\mathsf{eta}}$.

7 Other Results

It is quite easy to extend the method in [12] to prove confluence of all of the systems considered in this paper. (For the systems of inductive types see [9].)

One may also prove strong normalization of PFixT which is NPF with interleaving allowed. One simply always has to take least or greatest fixed-points of the functions Ψ_I or Ψ_E. I believe that one may construct an embedding out of this proof which finally embeds PFixT into NPF.

As was promised at the end of section 3 we finish by discussing Mendler's [10] system. It is an extension of system F by some kind of inductive and coinductive types. We shall concentrate on the inductive types: $\mu\alpha\rho$ is in the system if α occurs only positively in ρ. Interleaving is allowed. The term definition is extended by the usual rule (μ-I) and some eliminiation rule:

(μ-E) If $r^{\mu\alpha\rho}$ is a term and $s^{\forall\alpha.(\alpha\to\mu\alpha\rho)\to(\alpha\to\sigma)\to\rho\to\sigma}$ is a term, then $(r\mathrm{E}_\mu s)^\sigma$ is a term.

(β_μ) is replaced by some kind of recursion rule

$$(\beta_\mu)'' \qquad (\mathrm{C}_{\mu\alpha\rho}t)\mathrm{E}_\mu s \mapsto s(\mu\alpha\rho)(\lambda y^{\mu\alpha\rho}y)(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu s)t\ .$$

Mendler does not give any explanation of this rule but in [13] one can find one and also much more discussion in [9]. It turns out that strong normalization does not need the restriction to positivity which is however exploited in the proof in [10]. Let us sketch an embedding of the system without positivity requirement into NPF (whence the system inherits strong normalization from NPF): Set

$$(\mu\alpha\rho)' := \mu\beta\forall\gamma.(\forall\alpha.(\alpha\to\beta)\to(\alpha\to\gamma)\to\rho'\to\gamma)\to\gamma\ ,$$

$$(\mathrm{C}_{\mu\alpha\rho}t)' := \mathrm{C}_{(\mu\alpha\rho)'}\Big(\Lambda\gamma\lambda z^{\tau}.z(\mu\alpha\rho)'(\lambda y^{(\mu\alpha\rho)'}y)(\lambda x^{(\mu\alpha\rho)'}.x\mathrm{E}_{\mu}\gamma z)t'\Big)$$

with $\tau := \forall\alpha.(\alpha \to (\mu\alpha\rho)') \to (\alpha \to \gamma) \to \rho' \to \gamma$ and

$$(r\mathrm{E}_{\mu}s)' := r'\mathrm{E}_{\mu}\sigma' s' \ .$$

Following the procedure of section 6 one can indeed show that this gives an embedding.

Acknowledgements to Thorsten Altenkirch for many fruitful discussions and to the referees for their numerous hints.

References

1. Martín Abadi and Marcelo Fiore. Syntactic Considerations on Recursive Types. In *Proceedings of the Eleventh Annual IEEE Symposium on Logic in Computer Science*, pp. 242–252. IEEE Computer Society, July 1996.
2. Thorsten Altenkirch. Strong Normalization for T+. Unpublished note, 1993.
3. Henk Barendregt. Lambda Calculi with Types. In S. Abramsky, D. Gabbay and T. Maibaum, editors. Volume 2 of *Handbook of Logic in Computer Science*, pp. 117–309. Oxford Univ. Press, 1993.
4. Herman Geuvers. Inductive and Coinductive Types with Iteration and Recursion. In B. Nordström, K. Pettersson and G. Plotkin, editors. *Preliminary Proceedings of the Workshop on Types for Proofs and Programs, Båstad, June 1992*, pp. 193–217 (`ftp://ftp.cs.chalmers.se/pub/cs-reports/baastad.92/proc.dvi.Z`).
5. Jean-Yves Girard, Yves Lafont and Paul Taylor. Proofs and Types. Cambridge Univ. Press, 1989.
6. Carl A. Gunter. Semantics of Programming Languages: Structures and Techniques. Foundations of Computing Series. The MIT Press, Cambridge, Massachusetts, 1992.
7. Daniel Leivant. Contracting Proofs to Programs. In P. Odifreddi, editor. *Logic in Computer Science*, Vol. 31 APIC Studies in Data Processing, Academic Press, pp. 279–327, 1990.
8. Ralph Loader. Normalisation by Translation. Unpublished note announced on the "types" mailing list on April 6, 1995.
9. Ralph Matthes. Extensions of System F by Iteration and Primitive Recursion on Monotone Inductive Types. PhD thesis, University of Munich, 1998, to appear (available via `http://www.tcs.informatik.uni-muenchen.de/~matthes/`).
10. Nax P. Mendler. Recursive Types and Type Constraints in Second-Order Lambda Calculus. In *Proceedings of the Second Annual IEEE Symposium on Logic in Computer Science*, pp. 30–36. IEEE Computer Society, June 1987.
11. Femke van Raamsdonk and Paula Severi. On Normalisation. Technical Report CS-R9545, CWI, June 1995.
12. Masako Takahashi. Parallel Reduction in λ-Calculus, *Information and Computation* 118, pp. 120–127, 1995.
13. Tarmo Uustalu and Varmo Vene. A Cube of Proof Systems for the Intuitionistic Predicate μ-, ν-Logic. In M. Haveraaen and O. Owe, editors, *Selected Papers of the 8th Nordic Workshop on Programming Theory (NWPT '96), Oslo, Norway, December 1996*, volume 248 of *Research Reports, Department of Informatics, University of Oslo*, pp. 237–246, May 1997.