Monotone Fixed-Point Types And Strong Normalization

Ralph Matthes*
matthes@informatik.uni-muenchen.de
Lehr- und Forschungseinheit für theoretische Informatik
Institut für Informatik der Universität München, Germany

April 16, 1998

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. Due to the presence of F's impredicativity it is also possible to embed monotone inductive types (with full primitive recursion) into the system of non-interleaved positive fixed-point types. In the author's view this gives the easiest way to proving strong normalization for systems of inductive types.

1 Definition of the systems of monotone fixed-point types

We consider extensions of system F (see e.g. [GLT89]) 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 [GLT89], i.e. without contexts and with fixed types. 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

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

*I am thankful for support by the Volkswagenstiftung.

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 $r_1 \to r_2 \to r_3 \ldots$ (In [Mat98] 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. [AF96] (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 $(C_{\mu\alpha\rho}t)^{\mu\alpha\rho}$ 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

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

Because of this rule and its apparent similarity to retracts in domain theory the fixed-point types get the name retract types in [Geu93].
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 EMFixT of elimination-based monotone fixed-point types) or by assuming it in the elimination rule (thus defining the system IMFixT of introduction-based monotone fixed-point types).
Hence, the terms of EMFixT 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 $(C_{\mu\alpha\rho}mt)^{\mu\alpha\rho}$ 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 EMFixT by

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

Similarly, the terms of IMFixT 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 $(C_{\mu\alpha\rho}t)^{\mu\alpha\rho}$ is a term.

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

The relation $\mapsto$ of system F is extended for IMFixT by

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

Lemma 1 EMFixT and IMFixT are not normalizing.

Proof We only show it for EMFixT because the proof for IMFixT 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

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

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

We observe that the monotonicity witness does not enter the definition of beta reduction and therefore cannot ensure normalization in case of trivially monotone fixed-point types.
Note that with recursive types in type assignment systems it would not be possible to carry monotonicity witnesses around, and since [Men87] it is known that non-positivity always leads to the typability of non-normalizing terms.
In order to solve this problem in our setting of monotonicity witnesses we define the weak version wEMFixT of EMFixT by changing the definition of reduction to

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

Similarly, we define the weak version wIMFixT of IMFixT by replacing (β_{μ}) by

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

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

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

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 [Gun92], 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 wEMFixT and wIMFixT are strongly normalizing.

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

2 A system of non-interleaved positive fixed-point types

The system NIPFixT will have the standard (β_{μ})-rule $(C_{\mu\alpha\rho}t)\mathrm{E}_{\mu} \mapsto t$ 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 NIPosTypes of non-interleaved positive types and simultaneously for every $\rho \in \mathsf{NIPosTypes}$ the set $\mathsf{NIPos}(\rho)$ 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 $\mathsf{NINeg}(\rho)$ 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) $\alpha \in \mathsf{NIPosTypes}$ and $\mathsf{NIPos}(\alpha) := \mathsf{Typevars}$ and $\mathsf{NINeg}(\alpha) := \mathsf{Typevars} \setminus \{\alpha\}$.

($\times$) If $\rho \in \mathsf{NIPosTypes}$ and $\sigma \in \mathsf{NIPosTypes}$, then $\rho \times \sigma \in \mathsf{NIPosTypes}$.
$\mathsf{NIPos}(\rho \times \sigma) := \mathsf{NIPos}(\rho) \cap \mathsf{NIPos}(\sigma)$ and $\mathsf{NINeg}(\rho \times \sigma) := \mathsf{NINeg}(\rho) \cap \mathsf{Neg}(\sigma)$.

($\to$) If $\rho \in \mathsf{NIPosTypes}$ and $\sigma \in \mathsf{NIPosTypes}$, then $\rho \to \sigma \in \mathsf{NIPosTypes}$.
$\mathsf{NIPos}(\rho \to \sigma) := \mathsf{NINeg}(\rho) \cap \mathsf{NIPos}(\sigma)$ and $\mathsf{NINeg}(\rho \to \sigma) := \mathsf{NIPos}(\rho) \cap \mathsf{NINeg}(\sigma)$.

($\forall$) If $\rho \in \mathsf{NIPosTypes}$, then $\forall\alpha\rho \in \mathsf{NIPosTypes}$.
$\mathsf{NIPos}(\forall\alpha\rho) := \mathsf{NIPos}(\rho) \cup \{\alpha\}$ and $\mathsf{NINeg}(\forall\alpha\rho) := \mathsf{NINeg}(\rho) \cup \{\alpha\}$.

(μ) If $\rho \in \mathsf{NIPosTypes}$ and $\alpha \in \mathsf{NIPos}(\rho)$, then $\mu\alpha\rho \in \mathsf{NIPosTypes}$.
$\mathsf{NIPos}(\mu\alpha\rho) := \mathsf{NIPos}(\rho) \setminus \mathsf{FV}(\mu\alpha\rho)$ and $\mathsf{NINeg}(\mu\alpha\rho) := (\mathsf{NINeg}(\rho) \setminus \mathsf{FV}(\rho)) \cup \{\alpha\}$.

Let NIPFixT be the extension of system F to the types in NIPosTypes and with the standard rules (μ-I), (μ-E) and (β_μ) mentioned first. (The definitions for system F are still correct because NIPosTypes is closed under substitution which is somewhat awkward to prove because one needs additional statements on NIPos and NINeg to be proved simultaneously.)
Interleaving μ-types are ruled out in NIPFixT but not nesting! E.g. we may 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:

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

I consider NIPFixT to be a minor variant of F_{ret} in [Geu93] which is shown to be strongly normalizing by an embedding into Mendler's system [Men87] via a system of non-interleaved positive inductive types with primitive recursion (in [Geu93]).
NIPFixT is a subsystem of the system M2Lμ in [Lei90] (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 [Mat98] I give a direct proof of strong normalization of the variant of Mendler's system which simply leaves out the positivity requirement in [Men87] (which is crucial to the normalization proof presented there). I also show that this more liberal system embeds in a reduction-preserving way into NIPFixT. Therefore I think that a direct proof of strong normalization for NIPFixT would give an easier access to the above-mentioned normalization results.

Theorem 2 The system NIPFixT is strongly normalizing.

Before looking at the proof we try to see in which way NIPFixT embeds into the systems of monotone fixed-point types.

3 Embedding NIPFixT into monotone fixed-point types

Clearly, non-interleaved fixed-point types have to be monotone fixed-point types. This will be studied more carefully in this section.

3.1 Definition of monotonicity witnesses for NIPFixT

We want to define $\mathsf{map}_{\lambda\alpha\rho}$ as a closed term in NIPFixT of type $\forall\alpha\forall\beta.(\alpha \to \beta) \to \rho \to \rho[\alpha := \beta]$ for every $\mu\alpha\rho \in \mathsf{NIPosTypes}$, i.e. $\rho \in \mathsf{NIPosTypes}$ and $\alpha \in \mathsf{NIPos}(\rho)$. Because positivity and

negativity are defined simultaneously, we define auxiliary terms $\mathsf{comap}_{\lambda\alpha\rho}$ for $\alpha \in \mathsf{NINeg}(\rho)$ simultaneously:
Define for every type $\rho \in \mathsf{NIPosTypes}$ and every $\alpha \in \mathsf{NIPos}(\rho)$ (and $\beta \notin \{\alpha\} \cup \mathsf{FV}(\rho)$) a closed term

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

and for every $\alpha \in \mathsf{NINeg}(\rho)$ (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.
In order to get terms in normal form and not too much notation we will set

$$\mathsf{map}_{\lambda\alpha\rho} := \Lambda\alpha\Lambda\beta\lambda f^{\alpha\to\beta}\lambda x^{\rho}\underline{\mathsf{map}}_{\lambda\alpha\rho,\alpha,\beta}[f^{\alpha\to\beta}, x^{\rho}] \quad \text{and}$$

$$\mathsf{comap}_{\lambda\alpha\rho} := \Lambda\alpha\Lambda\beta\lambda f^{\alpha\to\beta}\lambda x^{\rho[\alpha:=\beta]}\underline{\mathsf{comap}}_{\lambda\alpha\rho,\alpha,\beta}[f^{\alpha\to\beta}, x^{\rho[\alpha:=\beta]}],$$

where $f \in \mathsf{Vars}$, $\underline{\mathsf{map}}_{\lambda\alpha\rho,\pi,\sigma}$ is a simultaneously defined term of type $\rho[\alpha := \sigma]$ with

$$x^{\rho[\alpha:=\pi]} \in \mathsf{FV}(\underline{\mathsf{map}}_{\lambda\alpha\rho,\pi,\sigma}) \subseteq \{f^{\pi\to\sigma}, x^{\rho[\alpha:=\pi]}\}$$

and $\underline{\mathsf{comap}}_{\lambda\alpha\rho,\pi,\sigma}$ is a simultaneously defined term of type $\rho[\alpha := \pi]$ with

$$x^{\rho[\alpha:=\sigma]} \in \mathsf{FV}(\underline{\mathsf{comap}}_{\lambda\alpha\rho,\pi,\sigma}) \subseteq \{f^{\pi\to\sigma}, x^{\rho[\alpha:=\sigma]}\}$$

and

$$\underline{\mathsf{map}}_{\lambda\alpha\rho,\pi,\sigma}[r^{\pi\to\sigma}, s^{\rho[\alpha:=\pi]}] := \underline{\mathsf{map}}_{\lambda\alpha\rho,\pi,\sigma}[f^{\pi\to\sigma}, x^{\rho[\alpha:=\pi]} := r, s] \quad \text{and}$$

$$\underline{\mathsf{comap}}_{\lambda\alpha\rho,\pi,\sigma}[r^{\pi\to\sigma}, s^{\rho[\alpha:=\sigma]}] := \underline{\mathsf{map}}_{\lambda\alpha\rho,\pi,\sigma}[f^{\pi\to\sigma}, x^{\rho[\alpha:=\sigma]} := r, s].$$

In the definitions we do not write the types of the variables f and x because the only allowed free variables are meant.
(triv) Case $\alpha \notin \mathsf{FV}(\rho)$. Then $\alpha \in \mathsf{NIPos}(\rho) \cap \mathsf{NINeg}(\rho)$. Set $\underline{\mathsf{map}}_{\lambda\alpha\rho,\pi,\sigma} := x$, $\underline{\mathsf{comap}}_{\lambda\alpha\rho,\pi,\sigma} := x$.
All the other cases are under the proviso "otherwise".
(V) Case $\rho = \alpha$. Then $\alpha \in \mathsf{NIPos}(\rho) \setminus \mathsf{NINeg}(\rho)$. Set $\underline{\mathsf{map}}_{\lambda\alpha\rho,\pi,\sigma} := fx$.
($\times$) Case $\rho = \rho_1 \times \rho_2$. If $\alpha \in \mathsf{NIPos}(\rho) = \mathsf{NIPos}(\rho_1) \cap \mathsf{NIPos}(\rho_2)$ then set

$$\underline{\mathsf{map}}_{\lambda\alpha\rho,\pi,\sigma} := \left\langle \underline{\mathsf{map}}_{\lambda\alpha\rho_1,\pi,\sigma}[f, x\mathsf{L}], \underline{\mathsf{map}}_{\lambda\alpha\rho_2,\pi,\sigma}[f, x\mathsf{R}] \right\rangle.$$

If $\alpha \in \mathsf{NINeg}(\rho) = \mathsf{NINeg}(\rho_1) \cap \mathsf{NINeg}(\rho_2)$ then set

$$\underline{\mathsf{comap}}_{\lambda\alpha\rho,\pi,\sigma} := \left\langle \underline{\mathsf{comap}}_{\lambda\alpha\rho_1,\pi,\sigma}[f, x\mathsf{L}], \underline{\mathsf{comap}}_{\lambda\alpha\rho_2,\pi,\sigma}[f, x\mathsf{R}] \right\rangle.$$

($\to$) Case $\rho = \rho_1 \to \rho_2$. If $\alpha \in \mathsf{NIPos}(\rho) = \mathsf{NIPos}(\rho_2) \cap \mathsf{NINeg}(\rho_1)$ then set

$$\underline{\mathsf{map}}_{\lambda\alpha\rho,\pi,\sigma} := \lambda y^{\rho_1[\alpha:=\sigma]}.\underline{\mathsf{map}}_{\lambda\alpha\rho_2,\pi,\sigma}\left[f, x(\underline{\mathsf{comap}}_{\lambda\alpha\rho_1,\pi,\sigma}[f, y])\right].$$

If $\alpha \in \mathsf{NINeg}(\rho) = \mathsf{NINeg}(\rho_2) \cap \mathsf{NIPos}(\rho_1)$ then set

$$\underline{\mathsf{comap}}_{\lambda\alpha\rho,\pi,\sigma} := \lambda y^{\rho_1[\alpha:=\pi]}.\underline{\mathsf{comap}}_{\lambda\alpha\rho_2,\pi,\sigma}\left[f, x(\underline{\mathsf{map}}_{\lambda\alpha\rho_1,\pi,\sigma}[f, y])\right].$$

($\forall$) Case $\rho = \forall\gamma\tau$. Because of $\alpha \in \mathsf{FV}(\rho)$, $\alpha \neq \gamma$. Therefore, if $\alpha \in \mathsf{NIPos}(\rho)$ then $\alpha \in \mathsf{NIPos}(\tau)$. Set

$$\underline{\mathsf{map}}_{\lambda\alpha\rho,\pi,\sigma} := \Lambda\gamma.\underline{\mathsf{map}}_{\lambda\alpha\tau,\pi,\sigma}[f, x\gamma].$$

If $\alpha \in \mathsf{NINeg}(\rho)$ then $\alpha \in \mathsf{NINeg}(\tau)$. Set

$$\underline{\mathsf{comap}}_{\lambda\alpha\rho,\pi,\sigma} := \Lambda\gamma.\underline{\mathsf{comap}}_{\lambda\alpha\tau,\pi,\sigma}[f, x\gamma].$$

The proviso for Λ-abstraction is met as we may assume that $\gamma \notin \mathsf{FV}(\pi) \cup \mathsf{FV}(\sigma)$. In order to verify type correctness we assume the same and therefore have $(\forall\gamma\tau)[\alpha := \sigma] = \forall\gamma.\tau[\alpha := \sigma]$ and $(\forall\gamma\tau)[\alpha := \pi] = \forall\gamma.\tau[\alpha := \pi]$.
Note that there is no clause named (μ) because this case is already covered by (triv) due to the fact that no interleaving is allowed.

3.2 The embeddings

An embedding $-'$ of NIPFixT in EMFixT is now immediately defined by induction on the terms of NIPFixT setting

(μ-I) $(C_{\mu\alpha\rho}t)' := C_{\mu\alpha\rho}\mathsf{map}_{\lambda\alpha\rho}t'$.

and homomorphic rules for all the other term formers. This gives terms in EMFixT because $\mathsf{map}_{\lambda\alpha\rho}$ is built without (μ-I) and hence already is a term of EMFixT.
It is easy to see that if $r \to s$ in NIPFixT, then $r' \to s'$ in EMFixT. (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 NIPFixT into IMFixT.
Unfortunately, the two embeddings do not preserve (β_μ) in wEMFixT and wIMFixT, respectively. An easy solution to this problem is the addition of η-rules for $\times$, $\to$ and $\forall$ as usual:

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

(where $\mathsf{FTV}(r)$ is the set of free type variables of r).
Indicate the presence of these additional rules by the suffix +eta in the names of the systems.
Now it is easy to see that in NIPFixT+eta (and trivially also in wEMFixT+eta and wIMFixT+eta) we have that

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

(where $\to^*$ denotes the reflexive and transitive closure of $\to$).
Therefore we get that whenever $r \to s$ in NIPFixT, then $r' \to^+ s'$ in the systems wEMFixT+eta and wIMFixT+eta, respectively (where $\to^+$ denotes the transitive closure of $\to$). This entails that from strong normalization of wEMFixT+eta or wIMFixT+eta, we can infer strong normalization of NIPFixT.
But now we show directly that even NIPFixT+eta is strongly normalizing and derive strong normalization of wEMFixT+eta and wIMFixT+eta from this fact via an embedding.
We show even more:
Let NIPFixT+ETA be the system NIPFixT+eta extended by the η-rule for fixed-point types:

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

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

Theorem 3 The system NIPFixT+ETA is strongly normalizing.

4 A direct proof of strong normalization for NIPFixT+ETA

Every term and type in this section is taken from NIPFixT+ETA.

4.1 Head forms, the set SN and saturated sets

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

(V) $x^\rho \vec{s}$

($\times$-I) $\langle r, s\rangle$

($\times$-R) $\langle r, s\rangle \mathsf{L}\vec{s}$, $\langle r, s\rangle \mathsf{R}\vec{s}$

($\to$-I) $\lambda x^\rho r$

($\to$-R) $(\lambda x^\rho r)s\vec{s}$

($\forall$-I) $\Lambda\alpha r$

($\forall$-R) $(\Lambda\alpha r)\sigma\vec{s}$

(μ-I) $C_{\mu\alpha\rho}t$

(μ-R) $(C_{\mu\alpha\rho}t)\mathrm{E}_\mu\vec{s}$

The vector $\vec{s}$ denotes a list made up of the symbols L, R and E_μ and of terms and types such that e.g. $x^\rho\vec{s}$ 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 $x^\rho\vec{s}$ is a term and the terms in $\vec{s}$ are in SN, then $x^\rho\vec{s} \in \mathsf{SN}$.

($\times$-I) If $r \in \mathsf{SN}$ and $s \in \mathsf{SN}$, then $\langle r, s\rangle \in \mathsf{SN}$.

($\beta_\times$) If $r\vec{s} \in \mathsf{SN}$ and $s \in \mathsf{SN}$, then $\langle r, s\rangle\mathsf{L}\vec{s} \in \mathsf{SN}$.
If $s\vec{s} \in \mathsf{SN}$ and $r \in \mathsf{SN}$, then $\langle r, s\rangle\mathsf{R}\vec{s} \in \mathsf{SN}$.

($\to$-I) If $r \in \mathsf{SN}$, then $\lambda x^\rho r \in \mathsf{SN}$.

($\beta_\to$) If $(\lambda x^\rho r)s\vec{s}$ is a term, $s \in \mathsf{SN}$ and $r[x^\rho := s]\vec{s} \in \mathsf{SN}$, then $(\lambda x^\rho r)s\vec{s} \in \mathsf{SN}$.

($\forall$-I) If $\Lambda\alpha r$ is a term and $r \in \mathsf{SN}$, then $\Lambda\alpha r \in \mathsf{SN}$.

($\beta_\forall$) If $(\Lambda\alpha r)\sigma\vec{s}$ is a term and $r[\alpha := \sigma]\vec{s} \in \mathsf{SN}$, then $(\Lambda\alpha r)\sigma\vec{s} \in \mathsf{SN}$.

(μ-I) If $t \in \mathsf{SN}$, then $C_{\mu\alpha\rho}t \in \mathsf{SN}$.

(μ-R) If $(C_{\mu\alpha\rho}t)\mathrm{E}_\mu\vec{s}$ is a term and $t\vec{s} \in \mathsf{SN}$, then $(C_{\mu\alpha\rho}t)\mathrm{E}_\mu\vec{s} \in \mathsf{SN}$.

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. $\mathsf{SN} \subseteq \mathsf{sn}$ 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. □

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 [RS95] and [Loa95].
I want to stress that it is invisible in this approach that we deal with NIPFixT+ETA instead of NIPFixT—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.
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 $x^\rho \vec{s}$ is a term of type τ and the terms in $\vec{s}$ are in SN, then $x^\rho \vec{s} \in \mathcal{M}$.

($\beta_\times$) If $r\vec{s} \in \mathcal{M}$ and $s \in \mathsf{SN}$, then $\langle r, s\rangle \mathsf{L}\vec{s} \in \mathcal{M}$. If $s\vec{s} \in \mathcal{M}$ and $r \in \mathsf{SN}$, then $\langle r, s\rangle \mathsf{R}\vec{s} \in \mathcal{M}$.

($\beta_\to$) If $(\lambda x^\rho r)s\vec{s}$ is a term, $s \in \mathsf{SN}$ and $r[x^\rho := s]\vec{s} \in \mathcal{M}$, then $(\lambda x^\rho r)s\vec{s} \in \mathcal{M}$.

($\beta_\forall$) If $(\Lambda\alpha r)\sigma\vec{s}$ is a term and $r[\alpha := \sigma]\vec{s} \in \mathcal{M}$, then $(\Lambda\alpha r)\sigma\vec{s} \in \mathcal{M}$.

(β_μ) If $(C_{\mu\alpha\rho}t)\mathrm{E}_\mu\vec{s}$ is a term and $t\vec{s} \in \mathcal{M}$, then $(C_{\mu\alpha\rho}t)\mathrm{E}_\mu\vec{s} \in \mathcal{M}$.

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 [Mat98].)

4.2 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

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

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

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

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 $\mathcal{M} \in \mathsf{SAT}_{\mu\alpha\rho}$

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

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 $M_E(\mathcal{M}) \cap \mathsf{SN} \in \mathsf{SAT}_{\mu\alpha\rho}$.

Lemma 5 Let $\mathcal{M} \in \mathsf{SAT}_{\mu\alpha\rho}$. $\mathcal{M}$ 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 $\forall r \in \mathcal{M} . r\mathrm{E}_\mu \in \phi(\mathcal{M})$.

Lemma 6 $\mu_E(\phi)$ is a pre-fixed point of Ψ_I and $\mu_I(\phi)$ is a post-fixed point of Ψ_E.

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

(μ-I) $\forall t \in \phi(\mu(\phi)) . C_{\mu\alpha\rho} t \in \mu(\phi)$.

(μ-E) $\forall r \in \mu(\phi) . r\mathrm{E}_\mu \in \phi(\mu(\phi))$.

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

(V) $\mathsf{SC}_\alpha[\vec{\alpha} := \vec{\mathcal{P}}] := \mathsf{cl}_\alpha(\Lambda)$, if α does not occur in the list $\vec{\alpha}$. (Note that $\mathsf{cl}_\alpha(\Lambda) = \mathsf{SN}_\alpha$.)
$\mathsf{SC}_\alpha[\vec{\alpha} := \vec{\mathcal{P}}] := \mathcal{P}_i$, if i is the smallest index such that $\alpha_i = \alpha$.

($\times$) $\mathsf{SC}_{\rho\times\sigma}[\vec{\alpha} := \vec{\mathcal{P}}] := \mathsf{SC}_\rho[\vec{\alpha} := \vec{\mathcal{P}}] \times \mathsf{SC}_\sigma[\vec{\alpha} := \vec{\mathcal{P}}]$.

($\to$) $\mathsf{SC}_{\rho\to\sigma}[\vec{\alpha} := \vec{\mathcal{P}}] := \mathsf{SC}_\rho[\vec{\alpha} := \vec{\mathcal{P}}] \to \mathsf{SC}_\sigma[\vec{\alpha} := \vec{\mathcal{P}}]$.

($\forall$) $\mathsf{SC}_{\forall\alpha\rho}[\vec{\alpha} := \vec{\mathcal{P}}] := \forall(\Phi)$ with $\Phi = (\Phi_\sigma)_\sigma$, where $\Phi_\sigma : \mathsf{SAT}_\sigma \to \mathsf{SAT}_{\rho[\vec{\alpha},\alpha:=\vec{\sigma},\sigma]}$ is defined by $\Phi_\sigma(\mathcal{P}) := \mathsf{SC}_\rho[\vec{\alpha}, \alpha := \vec{\mathcal{P}}, \mathcal{P}]$ (We assume that $\alpha \notin \vec{\alpha} \cup \mathsf{FV}(\vec{\sigma})$).

(μ) $\mathsf{SC}_{\mu\alpha\rho}[\vec{\alpha} := \vec{\mathcal{P}}] := \mu(\phi)$, where $\phi : \mathsf{SAT}_{\mu\alpha\rho} \to \mathsf{SAT}_{\rho[\vec{\alpha},\alpha:=\vec{\sigma},\mu\alpha\rho]}$ is defined by $\phi(\mathcal{P}) := \mathsf{SC}_\rho[\vec{\alpha}, \alpha := \vec{\mathcal{P}}, \mathcal{P}]$ (with the same assumptions 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 [Mat98].)

5 Embedding wEMFixT and wIMFixT into NIPFixT

The embedding of wEMFixT is derived from analyzing a normalization proof by saturated sets which was carried out by using an idea already present in [Alt93]. The other embedding may be seen as a dualization of the argument for wEMFixT.
In order to see what happens it is convenient to introduce existential types into NIPFixT which can later be eliminated in a reduction-preserving way. Hence, we embed wEMFixT into NIPFixT+ex which is NIPFixT extended by types of the form $\exists\alpha\rho$ (more formally we have to extend the definition of NIPosTypes), by the term rules

($\exists$-I) If $t^{\rho[\alpha:=\tau]}$ is a term, then $(C_{\exists\alpha\rho,\tau}t)^{\exists\alpha\rho}$ 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

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

Define for every type ρ of wEMFixT a type ρ' of NIPFixT+ex with the only non-homomorphic rule (letting $\mathsf{Mon}(\lambda\alpha\rho) := \forall\alpha\forall\beta.(\alpha \to \beta) \to \rho \to \rho[\alpha := \beta]$ for any ρ, e.g. also for ρ'):

$$(\mu\alpha\rho)' := \mu\alpha\exists\beta.\mathsf{Mon}(\lambda\alpha\rho') \times (\beta \to \alpha) \times \rho'[\alpha := \beta]$$

Define for every term r^ρ of wEMFixT a term r' of type ρ' of NIPFixT+ex as follows (the other clauses are homomorphic):

$$(\mu\text{-I})\quad (C_{\mu\alpha\rho}mt)' := C_{(\mu\alpha\rho)'}C_{\exists\beta.\mathsf{Mon}(\lambda\alpha\rho')\times(\beta\to(\mu\alpha\rho)')\times\rho'[\alpha:=\beta],(\mu\alpha\rho)'}\langle\langle m', \lambda x^{(\mu\alpha\rho)'}x\rangle, t'\rangle$$

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

Do the same for wIMFixT (this time directly into NIPFixT):

$$(\mu\alpha\rho)' := \mu\alpha\forall\beta.\mathsf{Mon}(\lambda\alpha\rho')\to(\alpha\to\beta)\to\rho'[\alpha:=\beta]$$

$$(\mu\text{-I})\quad (C_{\mu\alpha\rho}t)' := C_{(\mu\alpha\rho)'}\Big(\Lambda\beta\lambda z^{\mathsf{Mon}(\lambda\alpha\rho')}\lambda y^{(\mu\alpha\rho)'\to\beta}.z(\mu\alpha\rho)'\beta yt'\Big)$$

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

Some calculation will show that everything is type-correct (especially that only non-interleaved positive types are used!) and that reduction is preserved.
It is clear that the embeddings are also embeddings of wEMFixT+eta and wIMFixT+eta into NIPFixT+eta.

6 Embedding monotone inductive types into NIPFixT

As with monotone fixed-point types there are two possibilities to introduce the monotonicity witness: in the introduction rule giving the elimination-based system of monotone inductive types EMIT and alternatively in the elimination rules (one for iteration and one for primitive recursion) giving the elimination-based system of monotone inductive types IMIT.
I only give the definition of EMIT. The term rules are:

(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 $(C_{\mu\alpha\rho}mt)^{\mu\alpha\rho}$ is a term.

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

(μ-E$^+$) If $r^{\mu\alpha\rho}$ is a term and $s^{\rho[\alpha:=\mu\alpha\rho\times\sigma]\to\sigma}$ is a term, then $r\mathrm{E}^+_\mu\sigma s$ is a term.

The rule (μ-I) is as for EMFixT, the rule (μ-E) allows for iteration and (μ-E$^+$) gives full primitive recursion. The β-rules are as follows:

$$(\beta_\mu)\qquad (C_{\mu\alpha\rho}mt)\mathrm{E}_\mu\sigma s \mapsto s\Big(m(\mu\alpha\rho)\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma s)t\Big)$$

$$(\beta^+_\mu)\qquad (C_{\mu\alpha\rho}mt)\mathrm{E}^+_\mu\sigma s \mapsto s\Big(m(\mu\alpha\rho)(\mu\alpha\rho\times\sigma)\Big(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma s)x\rangle\Big)t\Big)$$

(In [Mat98] EMIT and IMIT are defined and studied.)

Theorem 4 There are reduction-preserving embeddings of EMIT and IMIT into NIPFixT. Hence, EMIT and IMIT are strongly normalizing.

Proof As in the previous section the embeddings may be read off the proofs of strong normalization by saturated sets. For IMIT this proof is done with the trick which was used for wEMFixT, for EMIT one has to dualize and hence use the same method as for wIMFixT. □

I would e.g. like to understand why IMIT needs the same idea as wEMFixT and not the same as wIMFixT.

7 Other results

It is quite easy to extend the method in [Tak95] to prove confluence of all of the systems considered in this paper. (For the systems of inductive types see [Mat98].)
One may also prove strong normalization of PFixT which is NIPFixT 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 NIPFixT. There is also an embedding of NIPFixT into the system of non-interleaved positive inductive types with primitive recursion and η-rules for $\times$, $\to$ and $\forall$. The main idea is to show a restricted form of functoriality of the canonical monotonicity witnesses which come with the system of non-interleaved positive inductive types.

References

[AF96] 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.

[Alt93] Thorsten Altenkirch. Strong normalization for T+. Unpublished note, 1993.

[Geu93] 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`).

[GLT89] Jean-Yves Girard, Yves Lafont and Paul Taylor. Proofs and types. Cambridge Univ. Press, 1989.

[Gun92] Carl A. Gunter. Semantics of programming languages: structures and techniques. Foundations of Computing Series. The MIT Press, Cambridge, Massachusetts, 1992.

[Lei90] 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.

[Loa95] Ralph Loader. Normalisation by translation. Unpublished note announced on the "types" mailing list on April 6, 1995.

[Mat98] 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 the homepage `http://www.tcs.informatik.uni-muenchen.de/~matthes/` after May 4).

[Men87] 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.

[RS95] Femke van Raamsdonk and Paula Severi. On normalisation. Technical Report CS-R9545, CWI, June 1995.

[Tak95] Masako Takahashi. Parallel Reduction in λ-Calculus, *Information and Computation* 118, pp. 120–127, 1995.