Extensions of System F by Iteration and Primitive Recursion on Monotone Inductive Types

Dissertation
an der Fakultät für Mathematik und Informatik
der Ludwig-Maximilians-Universität München

vorgelegt von
Ralph Matthes

eingereicht am 20. Mai 1998

1. Berichterstatter: Prof. Dr. H. Schwichtenberg
2. Berichterstatter: Prof. Dr. W. Buchholz
Tag der mündlichen Prüfung: 2. Februar 1999

In memoriam
Gerhard Schieferstein
1923–1992

Der mich ernstgenommen hat als Kind von elf Jahren.
Der mich die Querflöte spielen lehrte.
Der so meine Begeisterung für die klassische Musik entfachte.
Der mich aufschloß für die Welt der Technik, der Wissenschaft und ganz allgemein für alles das, wofür das Ringen um Einsicht und Erkenntnis sich lohnt.

Contents

Please refer to the index only in case the above table does not show the concept sought after.

Acknowledgements

Prof. Dr. Helmut Schwichtenberg was the man to introduce me to mathematical logic in 1990 and to direct my interest to natural deduction and to proofs of strong normalization for typed lambda calculi.
He made me share his[1] fascination for the idea of program extraction from constructive proofs. It seems that he never became tired of the ever-evolving proofs of strong normalization of systems with inductive types I presented to him over the last years.
I am grateful to him for the group of researchers he has formed in Munich, the guests he has attracted and generally the inspiring environment he has provided for me.

I thank all those who were part of this environment.

There are three colleagues who have been most important for me: Ulrich Berger, Thorsten Altenkirch and Felix Joachimski. I am very much indebted to the first two for all their invaluable suggestions, their comments and their unpublished manuscripts whereas I gratefully acknowledge the wonderful symbiosis which I enjoyed several times with Felix.

Although Prof. Dr. Wilfried Buchholz did not succeed in making me write my thesis in the field of classical proof theory (= ordinal analysis) I had the chance to learn from him that unconditional commitment to mathematical precision eventually pays off.

Nearly all of the thesis was written at my new "scientific home" at the chair for theoretical computer science of Prof. Dr. Peter Clote. It is my feeling that without his decision for me and the continual gentle pressure which followed I would have further delayed writing down my thesis. I like the new environment very much and therefore thank him and all the people at the chair (including Thorsten Altenkirch whom I mentioned above).

I would like to thank my parents, my sister and my friends for their strong support and constant encouragement. It was amazing how they reacted to my periods of mathematical joy and inspiration and to the times of stagnancy and mathematical infertility. I challenged their emotional capabilities—all the more as my mathematical activity kept being a secret to most of them—and I was rewarded a lot, sometimes unpleasant, sometimes surprising, but always interesting.

This research was partially supported by the Graduiertenkolleg "Sprache, Information, Logik" at the Ludwig-Maximilians-Universität München (financed by the Deutsche Forschungsgemeinschaft). I am also very thankful for support by the Volkswagenstiftung.

[1] See e. g. [BBS$^+$98].

Chapter 1

Introduction

We study systems of typed terms extending system F [Gir72, Rey74] with notions of β-reduction for every type construction. We focus on types of the form $\mu\alpha\rho$ whose intended interpretation is least pre-fixed-points of monotone operators. There are always two associated β-rules: The rule modelling iteration and the rule which models primitive recursion on the data structure. It is well known that data types such as the natural numbers may be represented in system F. This works very well for the definition of functions by iteration. But even the defined predecessor on natural numbers inside system F does not behave very well: The predecessor of the successor of n can be reduced to n only for numerals n and this process needs $O(n)$ steps. I do not know whether there is a proof that no reduction preserving embedding of primitive recursion on positive inductive types into system F is possible. But nobody seems to know such an embedding. Therefore, I decided to compare extensions of F allowing for primitive recursion on inductive types.

1.1 Motivation

This work is in some sense a chapter of the theory of complete lattices. But without the magnifying glasses of proof theory it would be quite short a chapter and moreover it would not contain any new result—new relative to Tarski's fixed-point theorem [Tar55][1].

First we use the magnifier of intuitionism to explore whether the classical results may also be proved constructively or whether they may be turned into statements ("constructivizations") permitting a constructive proof. Along this line we may define the term systems which are typed lambda calculi and also the encodings between them.

But this thesis is dominated by the question of the behaviour of term rewrite rules[2] of the systems and of their relation between systems. And this question may be turned into a problem of proof theory by using the Curry-Howard isomorphism [How80] in its most trivial reading[3]: The types of the extensions of system F are the formulas of an intuitionistic second-order propositional logic and the typed terms are simply proof terms in a natural deduction formulation of this intuitionistic logic. A term proves its type and its free typed term variables are the assumptions used in the proof. The term rewrite rules become proof transformations and they are justified if they do not change the type (formula) of the term (proof) and have as free term variables (assumptions) at most those which were present in the original term (proof).

We continue by adopting the proof-theoretic language. The analysis of proofs is greatly simplified

[1]We read: "Most of the results contained in this paper were obtained in 1939."

[2]We only consider β-rules for all the type constructs.

[3]Many subtleties concerning Curry-Howard isomorphisms may be found in [Geu93].

if the proof transformation system is normalizing and confluent because then one may only consider proofs in normal form. The comparison of the head forms of proofs (e. g. Lemma 2.24) and the set of normal forms (e. g. the inductively defined set NF on p. 18) shows what is gained by normalization. Consistency is a typical problem which is solved by studying the normal forms.

Since [Tai75] it has been common to establish strong normalization if possible. Although already normalization furnishes consistency and hence needs means of proof going beyond the logical system (which is an extension of pure second-order propositional logic) it is a challenge also to prove strong normalization. There are numerous published variations on that proof. Nevertheless I present another proof of strong normalization for system F which in my view separates the difficulties even more—most prominently the absence of reduct analysis in any reasoning with saturated sets which are a variant of the reducibility candidates (see e. g. [GLT89]).

Up to now I only spoke about system F. Although I present it in great detail it is only the basis on which to build the extensions by inductive types. Proof theorists might lose their interest in this work because system F is already fully impredicative and an extension by inductive types seems to give only "sugar" to the system. This would in fact be true if we only studied iteration on these inductive types. But we also study (full) primitive recursion. This time it is an algorithmic motivation: Functions defined by primitive recursion may be more efficient than functions defined by iteration. I alluded to it at the very beginning of this chapter by giving the example of the predecessor on the naturals (which in that encoding works iteratively).

This algorithmic motivation is intimately connected to a proof-theoretic question: What is the appropriate logical system allowing to extract efficient programs out of existence proofs[4]? Extraction of typed lambda terms[5] (viewed as programs) from intuitionistic proofs may be done via the modified realizability interpretation (due to G. Kreisel; see the citations in [Ber95]). I mention two examples where modified realizability is used for the extraction of programs out of normalization proofs: In [Ber93] a normalization algorithm is extracted from a proof of strong normalization of simply typed lambda calculus following Tait's computability predicate method [Tai67]. [vdP96b] builds on [Ber93] and extracts bounds on reduction lengths for Gödel's T from a variant of the normalization proof by the Tait method.

In [Ber95] a modified realizability interpretation is given for a system of interleaving positive inductive types with iteration only. The main insight I got from it was the motivation of the reduction rules for positive inductive types from it: The soundness of modified realizability only holds up to some equational theory on the terms and the attempt to prove soundness shows which equational axioms have to be postulated. All these efforts are in vain if the resulting equality theory is not effective but fortunately the axioms may be turned into term rewrite rules which form a strongly normalizing and confluent term rewrite system having hence decidable intensional equality. Hence, we may indeed interpret the extracted terms as programs.

Although I believe that a modified realizability interpretation could be given for the systems presented in this thesis I decided to be very informal about program extraction. But one should see the study of the term systems as the first part of a project aiming at the constructive interpretation of monotone inductive definitions. And it is a big part because the proof of soundness of the realizability interpretation is not so much different from the proof of strong normalization. Most of the difference is the extension from propositional logic (the term system) to full predicate logic. And this extension is quite easy: As long as there are no dependent types in the framework one can only use the objects in a uniform way in the proofs.

[4]More precisely: Proofs of Π^0_2-formulas.

[5]In [Par92] (system TTR) untyped lambda terms are extracted and only the reduction of untyped lambda calculus exhibits the algorithmic content. Correctness is expressed by typing rules which also include positive recursive types. In contrast to that I add reduction rules for every new type construct.

Perhaps the second part would become big if not only soundness (which guarantees that the extracted program meets the specification expressed in the existential statement) were the goal. A project going beyond the soundness theorem for modified realizability would be to formalize the normalization proofs and not only extract embeddings between systems out of them but also have a meta-theorem assuring that these encodings are always respecting β-reduction, hence that the extracted encodings are indeed embeddings (cf. section 9.3).
What are monotone inductive types? They are motivated by Tarski's theorem from which I only use the following: Let $(U, \leq)$ be a complete lattice and $\Phi : U \to U$ be monotone. Then Φ has a least fixed point[6] $\mathsf{lfp}(\Phi)$ which is given by

$$\mathsf{lfp}(\Phi) = \bigwedge\{M \in U | \Phi(M) \leq M\}.$$

The minimality of $\mathsf{lfp}(\Phi)$ gives a principle of induction:

$(\mathsf{lfp}\text{-E})\ \ \Phi(M) \leq M \Rightarrow \mathsf{lfp}(\Phi) \leq M.$

This will motivate iteration. As a derived rule we get

$(\mathsf{lfp}\text{-E}^+)\ \ \Phi(\mathsf{lfp}(\Phi) \wedge M) \leq M \Rightarrow \mathsf{lfp}(\Phi) \leq M.$

I call this rule the principle of extended induction[7]. It motivates primitive recursion.
The idea is as follows: We depart from the explicit representation of $\mathsf{lfp}(\Phi)$ as an infimum (which for pedagogic reasons is studied in a system of infimum types) because its reduction behaviour is not satisfactory. Instead we directly represent $\mathsf{lfp}(\Phi)$ by a type $\mu\alpha\rho$ in a system with iteration and primitive recursion.
We may now give an answer to the question on an appropriate logical system allowing to extract "efficient" programs out of existence proofs: It has extended induction as a primitive notion.
Monotone inductive definitions also draw attention in classical proof theory. The content of [Fef82, Rat96, GRS97, Rat98, Rat99] is orthogonal to the topics of this thesis, though. Their results are mainly based on classical logic, and they are very much concerned about measuring the strength of the theories at hand which clearly forbids the use of full impredicativity as expressed by system F. In this thesis, we exclusively study intuitionistic systems extending system F. The problems solved all have to do with reduction behaviour which does not play a role in the cited papers. Due to the different positions concerning impredicativity they focus on the possibility of getting least fixed points via transfinite iteration of a monotone operator starting with the empty set (hence being an approach from below) whereas we build on Tarski's theorem guaranteeing that the impredicatively justified infimum of the pre-fixed-points (hence being an approach from above) is also a pre-fixed-point and hence that a least pre-fixed-point exists.
Even the focus on primitive recursion instead of iteration (especially in the case of non-strictly positive inductive types) itself seems to be a quest for more impredicativity because the step term in the recursive definition may refer to the inductively defined set (modelled by the inductive type) in its entirety.
How do we model monotone inductive types? We allow $\mu\alpha\rho$ to be a type for any type ρ and any type variable α but we only may form terms of this type if we know monotonicity. How do we know that $\lambda\alpha\rho$ is monotone? By having already constructed a term of type

[6] In fact we only use that we have a least pre-fixed-point. The fact that this least pre-fixed-point is also a fixed point is never used in the systems of this thesis because it does not fit with β-reduction.
[7] In mathematical practice it is quite common to make unconcious use of "extended induction" instead of the original induction principle. Take e.g. the squaring function on the reals and try to prove that it has only natural numbers as values for natural number arguments. (One may already assume that addition does not leave $\mathbb{N}$.)

$\forall\alpha\forall\beta.(\alpha \to \beta) \to \rho \to \rho[\alpha := \beta]$. This term is called monotonicity witness and may arbitrarily use term formation rules for inductive types, hence permitting any interleaving of inductive types.
Interleaving inductive types are those where a free parameter of a subexpression of the type of the form $\mu\alpha\rho$ is bound by μ. Nesting would only mean that a type of the form $\mu\alpha\rho$ may be a subexpression of a type $\mu\alpha'\rho'$ with α' not free in $\mu\alpha\rho$. One could also call the interleaving inductive types inductive types with essential use of parameters.
Most presentations (for me important are [Lei90, Geu92]) restrict to positive inductive types, hence to a restriction already enforcing monotonicity on the level of types[8]. Mendler's type-theoretic system (with dependent types and subset types) in [Men87b] has the notion of monotone inductive types, i.e., μ-types are introduced under the assumption that the monotonicity statement (expressed via subset types) has already been derived. Surprisingly, the monotonicity proof is not used in the reduction rule. But this becomes clear when recognizing that the elimination rule does not express definition by primitive recursion. It corresponds to a definition by primitive recursion on a set inductively defined via a "positivization" of the monotone operator expressed by the monotone inductive type. And this positivization even works for non-monotone operators explaining why the monotonicity proof could be left out.
Why monotone inductive types?

- We try to prove as general results as possible.
- Monotone inductive types have a meta-theory where proofs may be done by structural recursion. In the case of positive inductive types we have the phenomenon that syntactic material pertaining to the canonical monotonicity witnesses necessary for formulating the rules of iteration and primitive recursion "pops up" during reduction. Therefore e.g. in the final soundness theorem in the proof of strong normalization one has to use quite complicated measures of induction. Compare [JO97] in the case of "abstract data type systems" where on p. 380 we find a tremendously complicated induction measure which is needed because the syntactic material is not carried around[9].
- It is nice to have an apparatus with expressive means which allow to formulate theorems which are only true after imposing some restrictions. In section 4.7.1 I present a slight variation of a very reasonable system for which normalization fails.
- And there is a nice inductive type which is monotone but not positive (due to U. Berger).

However, a large part of this thesis is concerned with showing how to embed systems of monotone inductive types into systems of non-interleaving positive inductive types while preserving reduction.[10] The main idea is to define "positivizations" of an arbitrary operator. Let us give an example: In calculus the limit inferior lim inf of a sequence $(a_k)_{k\in\mathbb{N}}$ of real numbers is formed by setting

$$\liminf_{k\to\infty} a_k := \lim_{n\to\infty} \inf\{a_k | k \geq n\}.$$

The limit to the right always exists (it may be infinite) because the sequence $(\inf\{a_k | k \geq n\})_{n\in\mathbb{N}}$ is monotonically increasing. And for this work it is most important that the monotonicity may be read off syntactically: The argument n occurs positively in the expression $\inf\{a_k | k \geq n\}$.

[8] Those systems nicely embed into the proposed systems of monotone inductive types.
[9] No doubt, this is more user-friendly but the user interface may be built up after having shown normalization and confluence.
[10] Let henceforth an embedding be an encoding which preserves reduction, i.e., for any reduction step in the source system one can do at least one reduction step in the target system to get from the encoding of a term to the encoding of its reduct.

In a similar fashion we get the lower and the upper monotonization of any operator. The upper monotonization gives rise to an embedding of monotone inductive types into non-interleaving strictly positive inductive types including the existential quantifier. The lower monotonization motivates embeddings into non-interleaving positive inductive types without the existential quantifier.
The embeddings work because the monotonization of a monotone operator is the operator itself, but the reflection of this fact joins the monotonicity via a monotonicity witness to the monotonicity via positivity.
Two other notions of monotonicity enter the proofs of strong normalization. They are the semantic requirement that a monotonicity witness is element of the saturated set

$$\forall \mathcal{P} \forall \mathcal{Q}.(\mathcal{P} \to \mathcal{Q}) \to \Phi(\mathcal{P}) \to \Phi(\mathcal{Q})$$

for Φ a family of operators on saturated sets which is a strong monotonicity requirement for Φ and finally the standard notion of monotonicity for such a family of operators. Unfortunately, I have no intuition how to reason by mere monotonicity of such operators in order to establish strong normalization. The good news is: I never had to reason in this way because always monotonizations could be found which were syntactically monotone (i. e., positive) and had properties good enough to let the proofs of strong normalization go through.

1.2 Outline of results

Most of the work is done in order to establish the following

Theorem Define four classes of systems as follows.

I Systems without primitive recursion.

- F: the basic impredicative system with $\to$ and $\forall$.
- eF: F extended by 0, 1, $\times$ and $+$.
- IT: eF extended by infimum types $i\alpha\rho$.
- eF+ex: eF extended by $\exists$.
- varEMIT−rec: the variant of the system of elimination-based (elimination rules without monotonicity witness) monotone inductive types with no primitive recursion but iteration.
- IMIT−rec: the system of introduction-based (introduction rule without monotonicity witness) monotone inductive types with no primitive recursion but iteration.
- MeIT−rec: the system in the spirit of Mendler with no primitive recursion but iteration.
- coMeIT−rec: the dual to MeIT−rec.
- Any other system of inductive types with iteration but no primitive recursion.

II The systems in IIa and in IIb to be defined next.

IIa Strictly positive systems without $\exists$.

- SPIT: the system of (interleaving) strictly positive inductive types (with iteration and primitive recursion and map terms built without primitive recursion but iteration).

- NISPIT: the system of non-interleaving strictly positive inductive types (with iteration and primitive recursion and map terms built with neither iteration nor recursion).

IIb Classes of systems with selected monotone inductive types.

- ESMIT: systems of elimination-based (unrestricted use of $\mu^{(+)}$-elimination in map terms) selected monotone inductive types.
- ESMITit: those of ESMIT without primitive recursion in the definitions of map terms.
- ESMITrec: those of ESMIT without iteration in the definitions of map terms.
- ESMIT+ex: systems like ESMIT but including $\exists$.
- ISMIT: systems of introduction-based (unrestricted use of μ-introduction in map terms) selected monotone inductive types.
- ISMIT+ex: systems like ISMIT but including $\exists$.

III 20 other systems of inductive types.

- PIT=PITit: the system of positive inductive types with iteration and primitive recursion where interleaving and non-strict positivity are allowed and the map terms do not use primitive recursion but only iteration.
- PITrec: like PITit but the map terms use primitive recursion and no iteration.
- PIT+ex: PIT extended by $\exists$.
- SPIT+ex: SPIT extended by $\exists$.
- NIPIT: the system of non-interleaving (but including non-strictly) positive inductive types with iteration and primitive recursion and map terms built with neither iteration nor recursion.
- NISPIT+ex: NISPIT extended by $\exists$.
- EMIT: the system of elimination-based (elimination rules without monotonicity witness) monotone inductive types with primitive recursion and iteration.
- EMIT−it: EMIT without iteration.
- varEMIT: a variant of EMIT with a slightly weaker requirement of monotonicity for the monotonicity witnesses.
- IMIT: the system of introduction-based (introduction rule without monotonicity witness) monotone inductive types with primitive recursion and iteration.
- IMIT−it: IMIT without iteration.
- varIMIT: a variant of IMIT with a slightly weaker requirement of monotonicity for the monotonicity witnesses.
- IMIT+ex: IMIT extended by $\exists$.
- MeIT: the system in the spirit of Mendler ("Mendler's Inductive Types") with iteration and primitive recursion.
- MeIT−it: MeIT without iteration (nearer to Mendler's [Men87a] system).
- coMeIT: the dual (the explanation via "monotonizations" of operators is dual) to MeIT.
- coMeIT−it: coMeIT without iteration.

- UVIT: "Uustalu's and Vene's Inductive Types", a slight generalization of MeIT (which serves for clarifying the embedding of MeIT into NISPIT+ex) also with iteration and primitive recursion.
- MePIT: the subsystem of NISPIT+ex which is actually needed for embedding MeIT into NISPIT+ex. It also has iteration and primitive recursion.
- coMePIT: the subsystem of NIPIT which is actually needed for embedding coMeIT into NIPIT; with iteration and primitive recursion.

Let an embedding be an encoding which preserves reduction, i. e., for any reduction step in the source system one can do at least one reduction step in the target system to get from the encoding of a term to the encoding of its reduct.
Then the following holds: There are embeddings of each system in I into any other. There are embeddings of each system in II into any of III and each system of III embeds into any other system of III. Hence, the systems in I are embedding equivalent to each other, and the same holds for the systems in III.
Every system in I, II and III is strongly normalizing and confluent.

All of the above systems are motivated and defined carefully. Examples are given and a lot of additional information is taken out of the proofs, among them naïve reduction-free normalization, consistency, the impossibility of proving the Peirce formula, an insight how much easier weak normalization is than strong normalization and even embeddings into systems of fixed-point types.

In chapter 2 the systems without inductive types are formulated and analyzed very carefully. Starting from system F with arrow types and universal types only, we subsequently add 0, 1, product types, sum types, existential types and infimum types.
Chapter 3 shows the different notions of inductive types (including strictly positive and positive and non-interleaving positive inductive types) and measures for them. Examples are considered.
Term systems for monotone inductive types are studied in chapter 4. The two possibilities of fixing a monotonicity witness to the term rules are presented. There is a non-trivial example of monotonicity which does not come from positivity. Also the systems with selected monotone inductive types having a specified monotonicity witness for every inductive type are defined and embedded into the systems with monotone inductive types by using a notion of stratified term. An elaborate discussion on the relation of iteration and recursion is included. The iterative fragment of varEMIT is embedded into the system of infimum types by simply reflecting the proof of Tarski's theorem.
In chapter 5 systems with positive inductive types are defined which means to define the canonical monotonicity witnesses whose stratification and uniformity alone guarantee strong normalization. It is presented how much easier the definitions are for subsystems like that of strictly positive inductive types or of non-interleaving positive inductive types. Very special instances are MePIT and coMePIT which may be seen as examples for the calculation of canonical monotonicity witnesses.
Chapter 6 is the heart of this thesis: The junction between monotone inductive types and non-interleaving positive inductive is introduced in form of MeIT and coMeIT the first of which is a variant of Mendler's [Men87a] system and the second a dual to it. The introductory sections to that chapter are essential for understanding why the embeddings work.
In chapter 7 we collect the embeddings established so far and prove the statements on embeddings in the theorem above.
Confluence is proved by a variant to Takahashi's [Tak95] method in chapter 8.

Chapter 9 and appendix A contain the most technical part: The proofs of strong normalization for several systems. Again much effort has been invested to make the proofs as clear and as modular as possible. The variations on the proof for EMIT are put in the appendix although they show a wealth of subtleties. A final discussion explains how the embeddings of monotone inductive types into MePIT and coMePIT could have been found.
Appendix B discusses possible extensions which came to my mind during the writing of this thesis mostly concerning fixed-point types, η-reduction, functoriality of the monotonicity witnesses and permutative conversions for inductive types. A list of open problems is given.

As central points of this thesis I consider:

- Monotone inductive types formulated[11] and proved to be strongly normalizing and confluent.
- Selected monotone inductive types formulated (as abstraction from the canonical monotonicity witnesses for positive inductive types) and embedded into monotone inductive types via an inductively defined set of stratified terms.
- Mendler's system proved to be strongly normalizing without any restriction to forming inductive types.
- Formulation of a dual (coMeIT) to Mendler's system.
- Embedding of monotone inductive types into non-interleaving positive inductive types. The main step: varEMIT embeds into coMeIT and varIMIT embeds into MeIT. Explanation why such easy embeddings exist.
- Reduction of iteration to primitive recursion even for introduction-based monotone inductive types.
- A measure of height (h) for abstracted types which allows to define the canonical monotonicity witnesses for interleaving positive inductive types.
- A technical device: The vector notation for multiple eliminations.
- The separation of reduct analysis from any reasoning on saturated sets.
- The definition of infimum types to clarify the encoding of iteration on elimination-based monotone inductive types inside system F.
- The organization of the proof of strong normalization such that always an introduction-based and an elimination-based definition of computability predicates is possible and that embeddings may be read off the proof.
- The equivalence with respect to embeddings of the 20 systems of class III in the above theorem.

---

[11] The very idea of considering primitive recursion in the presence of some arbitrary term of "functorial strength" already occurs in [Alt93c] together with a sketch of an elimination-based normalization proof.

Chapter 2

The basic framework

System F and its extension by 0, 1, product and sum types are defined in all details. An extension by types expressing infima is proposed. Finally, the extension by existentially quantified types is studied.

2.1 System F

The following definitions define a variant of the historical system F (invented independently by Girard [Gir72] and Reynolds [Rey74]), which will be my starting point to investigate impredicative systems. A presentation which focuses more on the main ideas is to be found in [GLT89]. [Mit96] also contains an extensive discussion of polymorphism. Good references (among numerous other good presentations) are also [Bar93] and [Gal90].

2.1.1 Types

Assume a countably infinite set $\mathsf{Typevars}$ of symbols which are considered as type variables. By induction define the set Types of types as follows:

(V) If $\alpha \in \mathsf{Typevars}$, then $\alpha \in \mathsf{Types}$.

($\to$) If $\rho \in \mathsf{Types}$ and $\sigma \in \mathsf{Types}$, then $\rho \to \sigma \in \mathsf{Types}$.

($\forall$) If $\alpha \in \mathsf{Typevars}$ and $\rho \in \mathsf{Types}$, then $\forall\alpha\rho \in \mathsf{Types}$.

The variable α in $\forall\alpha\rho$ is considered to be bound by $\forall$—more precisely, every occurrence of α in ρ which is not bound deeper inside ρ is bound by $\forall\alpha$. We adopt the convention not to distinguish between types which only differ in the names of their bound variables, e. g. we simply identify the types $\forall\alpha.\gamma \to (\forall\beta.\alpha \to \beta)$ and $\forall\delta.\gamma \to (\forall\alpha.\delta \to \alpha)$, assuming $\alpha, \beta, \gamma, \delta \in \mathsf{Typevars}$. As in this example parentheses are used to indicate the inductive build-up of the types which was suppressed in the inductive definition because it will be understood in all the inductive definitions of concepts that the defining rules freely generate the structure[1]. The dot notation hides a pair of parentheses, which opens at the dot and closes as far to the right as is syntactically possible. Moreover, $\forall$ is assumed to bind stronger than $\to$, and iterated $\to$ is associated to the right (i. e., $\rho \to \sigma \to \tau$ means $\rho \to (\sigma \to \tau)$).
For every type ρ we define the finite set $\mathsf{FV}(\rho)$ of type variables occurring free in ρ by recursion on the inductive definition of the types ρ as follows:

[1] Not only on the meta-level every inductive definition is considered to be deterministic but also the inductive types studied in this thesis exclusively model deterministic inductive definitions which therefore allow (transfinite) recursion (see e. g. [Acz77, p. 744]).

(V) $\mathsf{FV}(\alpha) := \{\alpha\}$.

($\to$) $\mathsf{FV}(\rho \to \sigma) := \mathsf{FV}(\rho) \cup \mathsf{FV}(\sigma)$.

($\forall$) $\mathsf{FV}(\forall\alpha\rho) := \mathsf{FV}(\rho) \setminus \{\alpha\}$.

This definition is compatible with the renaming convention.
Define the resulting[2] type $\rho[\vec{\alpha} := \vec{\sigma}]$ of simultaneously substituting the finite list of types $\vec{\sigma}$ for the (equally long) list of type variables $\vec{\alpha}$ in a type ρ by recursion on ρ as follows:

(V) $\alpha[\vec{\alpha} := \vec{\sigma}] := \alpha$, if α does not occur in the list $\vec{\alpha}$.
$\alpha[\vec{\alpha} := \vec{\sigma}] := \sigma_i$, if i is the smallest index such that $\alpha_i = \alpha$. (Subscripts indicate the respective elements in the list.)

($\to$) $(\rho \to \sigma)[\vec{\alpha} := \vec{\sigma}] := \rho[\vec{\alpha} := \vec{\sigma}] \to \sigma[\vec{\alpha} := \vec{\sigma}]$.

($\forall$) $(\forall\alpha\rho)[\vec{\alpha} := \vec{\sigma}] := \forall\alpha.\rho[\vec{\alpha} := \vec{\sigma}]$, where according to the renaming convention we assume that α is different from the $\vec{\alpha}$ and does not occur free in any of the $\vec{\sigma}$ (thanks to the infinite supply of type variables).

This definition is again compatible with the renaming convention.
Due to the assumption in the rule ($\forall$) which will be written as $\alpha \notin \vec{\alpha} \cup \mathsf{FV}(\vec{\sigma})$ we have for $\alpha \neq \beta$

- $(\forall\alpha\alpha)[\alpha := \beta] \neq \forall\alpha\beta$
- $(\forall\alpha\beta)[\beta := \alpha] \neq \forall\alpha\alpha$

We adopt the naming convention that α, β, γ and δ (even if decorated e. g. with subscripts) always denote type variables, and ρ, σ and τ (also with decoration) denote types.

Lemma 2.1 If $\alpha \notin \mathsf{FV}(\rho)$, then $\rho[\vec{\alpha}, \alpha := \vec{\sigma}, \sigma] = \rho[\vec{\alpha} := \vec{\sigma}]$.

Proof Induction on ρ. □

Corollary 2.2 If $\vec{\alpha} \cap \mathsf{FV}(\rho) = \emptyset$ then $\rho[\vec{\alpha} := \vec{\sigma}] = \rho$. □

Lemma 2.3 $\mathsf{FV}(\rho[\vec{\alpha} := \vec{\sigma}]) = (\mathsf{FV}(\rho) \setminus \vec{\alpha}) \cup \bigcup\{\mathsf{FV}(\sigma_i) | \alpha_i \in \mathsf{FV}(\rho) \wedge \alpha_i \neq \alpha_j \text{ for } j < i\}$, which trivially implies $\mathsf{FV}(\rho[\vec{\alpha} := \vec{\sigma}]) \subseteq (\mathsf{FV}(\rho) \setminus \vec{\alpha}) \cup \mathsf{FV}(\vec{\sigma})$.

Proof Induction on ρ. □

Corollary 2.4 If $\alpha \notin \vec{\alpha} \cup \mathsf{FV}(\vec{\sigma})$ then $\alpha \in \mathsf{FV}(\rho) \iff \alpha \in \mathsf{FV}(\rho[\vec{\alpha} := \vec{\sigma}])$. □

Lemma 2.5 If $\alpha \notin \vec{\alpha} \cup \mathsf{FV}(\vec{\sigma})$ then $(\rho[\alpha := \sigma])[\vec{\alpha} := \vec{\sigma}] = (\rho[\vec{\alpha} := \vec{\sigma}])[\alpha := \sigma[\vec{\alpha} := \vec{\sigma}]]$.

Proof Induction on ρ using Corollary 2.2. □

Lemma 2.6 If $\alpha \notin \vec{\alpha} \cup \mathsf{FV}(\vec{\sigma})$ then $\rho[\vec{\alpha}, \alpha := \vec{\sigma}, \sigma] = (\rho[\vec{\alpha} := \vec{\sigma}])[\alpha := \sigma]$.

Proof Induction on ρ using Corollary 2.2. □

[2] Note that only the effect of substituting types for type variables in types is studied and not the type substitution itself as a finitely presented function (giving rise to a substitution calculus).

More on the renaming convention can be found in Barendregt's [Bar84, pp. 25–27, 577–581]. The convention there is to "identify α-congruent terms on a syntactic level" (p. 26) as is done here with types. The "variable convention" says that for finitely many objects which have bound variables and occur together in some mathematical context, it is always assumed that the bound variables are different from all of the free variables.
In this thesis, I do not follow Barendregt's variable convention but always make the comment about what we have to assume on the bound variables in order to have a concept correctly defined. The reason for this cautious approach is that in many cases it is not clear (to me) whether the mathematical context at hand contains finitely or infinitely many such objects. Moreover, sometimes it is useful to have the same variable free and bound in a clause of a definition (see e.g. the clause (β^+_μ) in section 4.1.1). Finally, I do not completely understand what the variable convention really says about a definition clause such as

$$(\forall)\quad \mathsf{FV}(\forall\alpha\rho) := \mathsf{FV}(\rho) \setminus \{\alpha\}.$$

Therefore my convention merely means that I consider types only up to α-equivalence.
In loc. cit. also the approach by de Bruijn [dB72] (and some approach by Curry) is presented. Another truly accurate treatment of the renaming convention would be to define α-equivalence and type substitution simultaneously by recursion on the complexity of the types as was e.g. carried out in [Rei96].
Note that the Curry-Howard isomorphism means on the type level that we see the type variables as propositional variables and the types as propositions. The types of system F "are" hence the formulas of second-order propositional logic with only universal quantification over propositions and implication.

2.1.2 Terms

A slightly erroneous term system is defined and then—by changing the variables to be more polymorphic—the system forming the core of all the extensions considered in this thesis.

A first attempt to define the terms and a variable convention

Assume for every type ρ pairwise disjoint countably infinite sets $\mathsf{Vars}(\rho)$ of symbols which are considered as term variables of type ρ. By simultaneous induction define the sets Λ_ρ of typed terms of system F of type ρ and simultaneously define for every term r in any Λ_ρ the finite set $\mathsf{FV}(r)$ of term variables occurring free in r (overloading the previously defined FV for types) as follows:

(V) If $x \in \mathsf{Vars}(\rho)$, then $x \in \Lambda_\rho$. $\mathsf{FV}(x) := \{x\}$.

($\to$-I) If $x \in \mathsf{Vars}(\rho)$ and $r \in \Lambda_\sigma$, then $\lambda xr \in \Lambda_{\rho\to\sigma}$. $\mathsf{FV}(\lambda xr) := \mathsf{FV}(r) \setminus \{x\}$.

($\to$-E) If $r \in \Lambda_{\rho\to\sigma}$ and $s \in \Lambda_\rho$, then $rs \in \Lambda_\sigma$. $\mathsf{FV}(rs) := \mathsf{FV}(r) \cup \mathsf{FV}(s)$.

($\forall$-I) If $r \in \Lambda_\rho$ and $\alpha \in \mathsf{Typevars}$, then $\Lambda\alpha r \in \Lambda_{\forall\alpha\rho}$, provided that $\alpha \notin \mathsf{FV}(\sigma)$ for any σ with $\mathsf{FV}(r) \cap \mathsf{Vars}(\sigma) \neq \emptyset$. $\mathsf{FV}(\Lambda\alpha r) := \mathsf{FV}(r)$.

($\forall$-E) If $r \in \Lambda_{\forall\alpha\rho}$ and $\sigma \in \mathsf{Types}$, then $r\sigma \in \Lambda_{\rho[\alpha:=\sigma]}$. $\mathsf{FV}(r\sigma) := \mathsf{FV}(r)$.

It should be noted that the proviso in the rule ($\forall$-I) is usually called the eigenvariable condition which becomes clear in the discussion of the Curry-Howard isomorphism at the end of this section.

This definition is again compatible with the renaming convention for bound type variables (the only critical case being ($\forall$-E)).
The term variable x in λxr is considered to be bound by λ, and the type variable α in $\Lambda\alpha r$ is considered to be bound by Λ. We again do not want to distinguish between terms which only differ in the names of bound variables (term and type variables). E.g. if α and β are different type variables and $x \in \mathsf{Vars}(\alpha)$ and $y \in \mathsf{Vars}(\beta)$, then $\lambda xx \in \Lambda_{\alpha\to\alpha}$ and $\lambda yy \in \Lambda_{\beta\to\beta}$. Therefore $\Lambda\alpha\lambda xx \in \Lambda_{\forall\alpha.\alpha\to\alpha}$ and $\Lambda\beta\lambda yy \in \Lambda_{\forall\beta.\beta\to\beta}$. Because of the convention on identifying bound type variables in types, we have $\forall\alpha.\alpha \to \alpha = \forall\beta.\beta \to \beta$ which by compatibility implies $\Lambda_{\forall\alpha.\alpha\to\alpha} = \Lambda_{\forall\beta.\beta\to\beta}$. How can one rename the bound type variable α in $\Lambda\alpha\lambda xx$ to β? This would be possible if one were allowed to also rename $x \in \mathsf{Vars}(\alpha)$ to $y \in \mathsf{Vars}(\beta)$. I think this would be quite hard to justify. But we do want to identify $\Lambda\alpha\lambda xx$ and $\Lambda\beta\lambda yy$. Therefore we abandon this definition.

The correct definition of terms which allows for a variable convention

Assume a countably infinite set Vars of symbols which are considered as term variables without a type. By simultaneous induction define the sets Λ_ρ of typed terms of system F of type ρ and simultaneously define for every term r in any Λ_ρ the finite set $\mathsf{FV}(r)$ of typed term variables occurring free in r (overloading the previously defined FV for types) as follows:

(V) If $x \in \mathsf{Vars}$, then $(x, \rho) \in \Lambda_\rho$. $\mathsf{FV}((x, \rho)) := \{(x, \rho)\}$.

($\to$-I) If $x \in \mathsf{Vars}$ and $r \in \Lambda_\sigma$, then $\lambda(x, \rho)r \in \Lambda_{\rho\to\sigma}$. $\mathsf{FV}(\lambda(x, \rho)r) := \mathsf{FV}(r) \setminus \{(x, \rho)\}$.

($\to$-E) If $r \in \Lambda_{\rho\to\sigma}$ and $s \in \Lambda_\rho$, then $rs \in \Lambda_\sigma$. $\mathsf{FV}(rs) := \mathsf{FV}(r) \cup \mathsf{FV}(s)$.

($\forall$-I) If $r \in \Lambda_\rho$ and $\alpha \in \mathsf{Typevars}$, then $\Lambda\alpha r \in \Lambda_{\forall\alpha\rho}$, provided that $\alpha \notin \mathsf{FV}(\sigma)$ for any σ for which there is an $x \in \mathsf{Vars}$ such that $(x, \sigma) \in \mathsf{FV}(r)$. $\mathsf{FV}(\Lambda\alpha r) := \mathsf{FV}(r)$.

($\forall$-E) If $r \in \Lambda_{\forall\alpha\rho}$ and $\sigma \in \mathsf{Types}$, then $r\sigma \in \Lambda_{\rho[\alpha:=\sigma]}$. $\mathsf{FV}(r\sigma) := \mathsf{FV}(r)$.

This definition is again compatible with the renaming convention for bound type variables (the only critical case again being ($\forall$-E)).
The untyped term variable x in $\lambda(x, \rho)r$ is considered to be bound by λ, and the type variable α in $\Lambda\alpha r$ is considered to be bound by Λ. (Again we mean every occurrence inside r which is not bound deeper inside r.) We again do not distinguish between terms which only differ in the names of bound variables (term and type variables). This will be referred to as the variable convention (which of course relies on the renaming convention for bound type variables in types). E.g. if α and β are different type variables and $x \in \mathsf{Vars}$ and $y \in \mathsf{Vars}$, then $\lambda(x, \alpha)(x, \alpha) \in \Lambda_{\alpha\to\alpha}$ and $\lambda(y, \beta)(y, \beta) \in \Lambda_{\beta\to\beta}$. Therefore $\Lambda\alpha\lambda(x, \alpha)(x, \alpha) \in \Lambda_{\forall\alpha.\alpha\to\alpha}$ and $\Lambda\beta\lambda(y, \beta)(y, \beta) \in \Lambda_{\forall\beta.\beta\to\beta}$. Because of the convention on identifying bound type variables in types, we have $\forall\alpha.\alpha \to \alpha = \forall\beta.\beta \to \beta$ which by compatibility implies $\Lambda_{\forall\alpha.\alpha\to\alpha} = \Lambda_{\forall\beta.\beta\to\beta}$. The variable convention for term variables without types allows us to identify $\lambda(x, \alpha)(x, \alpha)$ and $\lambda(y, \alpha)(y, \alpha)$ and then the type variable convention identifies $\Lambda\alpha\lambda(x, \alpha)(x, \alpha)$ and $\Lambda\beta\lambda(y, \beta)(y, \beta)$. Therefore we keep this definition.
The simultaneous definition of FV on terms is compatible with the variable convention thanks to the proviso in the rule ($\forall$-I), which is called the eigenvariable condition (as mentioned before).

Lemma 2.7 For every term r there is a unique type ρ with $r \in \Lambda_\rho$.

Proof Induction on r. □

As is usual we will use type superscripts for variables instead of the pair notation (x, ρ) and we also use them for terms instead of type subscripts as in Λ_ρ: From now on x^ρ is written instead of (x, ρ), and $r^\rho \in \Lambda$ is written instead of $r \in \Lambda_\rho$. If r happens to be a variable we further abbreviate to $r \in \Lambda$ thus avoiding twice the same superscript. Using these conventions we can restate the definition of terms (this time leaving out the definition of FV) as follows:

(V) If $x \in \mathsf{Vars}$, then $x^\rho \in \Lambda$.

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

($\to$-E) If $r^{\rho\to\sigma} \in \Lambda$ and $s^\rho \in \Lambda$, then $(rs)^\sigma \in \Lambda$.

($\forall$-I) If $r^\rho \in \Lambda$, then $(\Lambda\alpha r)^{\forall\alpha\rho} \in \Lambda$, provided that $\alpha \notin \mathsf{FV}(\sigma)$ for any σ such that there is an x^σ in $\mathsf{FV}(r)$.

($\forall$-E) If $r^{\forall\alpha\rho} \in \Lambda$, then $(r\sigma)^{\rho[\alpha:=\sigma]} \in \Lambda$.

Unfortunately "Λ" is used as the name of the set of terms and also as the symbol for type abstraction. Nevertheless ambiguities cannot arise.
If r is a term then $r : \rho$ is defined to be the statement $r \in \Lambda_\rho$. r^ρ may either be used as synonymous to $r : \rho$ or in place of r but implicitly introducing the unique ρ with $r : \rho$ or imposing a restriction on r if ρ has previously been introduced.
As for types we freely use parentheses to indicate the build-up of a term which again was not made explicit in the definition (as was promised in the section on types). The dot notation is also employed. Λ and λ are assumed to bind stronger than application, iterated term application (that is nested use of ($\to$-E)) is associated to the left (i. e., rst means $(rs)t$) which fits perfectly with the right-associative $\to$ for types.
If $r : \rho \to \sigma$, we say that r is of arrow type. If $r : \forall\alpha\rho$, we speak of r as of universal type.
We adopt the naming convention that x, y and z (also with decoration) always denote untyped term variables, and r, s and t denote terms.
For every term r we define the finite set $\mathsf{FTV}(r)$ of type variables occurring free in r by recursion on r as follows:

(V) $\mathsf{FTV}(x^\rho) := \mathsf{FV}(\rho)$.

($\to$-I) $\mathsf{FTV}(\lambda x^\rho r) := \mathsf{FV}(\rho) \cup \mathsf{FTV}(r)$.

($\to$-E) $\mathsf{FTV}(rs) := \mathsf{FTV}(r) \cup \mathsf{FTV}(s)$.

($\forall$-I) $\mathsf{FTV}(\Lambda\alpha r) := \mathsf{FTV}(r) \setminus \{\alpha\}$.

($\forall$-E) $\mathsf{FTV}(r\sigma) := \mathsf{FTV}(r) \cup \mathsf{FV}(\sigma)$.

This definition is compatible with the variable convention.

Lemma 2.8 $\mathsf{FV}(\rho) \subseteq \mathsf{FTV}(r^\rho)$.

Proof Induction on r. For the rule ($\forall$-E) we need Lemma 2.3. □

Lemma 2.9 $x^\rho \in \mathsf{FV}(r) \Rightarrow \mathsf{FV}(\rho) \subseteq \mathsf{FTV}(r)$.

Proof Induction on r. For the rule ($\forall$-I) we use the proviso. □

Define the resulting term $r[\vec{x}^{\vec{\rho}} := \vec{s}]$ of simultaneously substituting the finite list of terms $\vec{s}^{\vec{\rho}}$ (i. e., $s_i : \rho_i$) for the (equally long) list of typed variables $\vec{x}^{\vec{\rho}}$ (i. e., $x_i : \rho_i$) in r by recursion on r and simultaneously prove that $r^\rho[\vec{x}^{\vec{\rho}} := \vec{s}] : \rho$ and that (compare Lemma 2.3) $\mathsf{FV}(r[\vec{x}^{\vec{\rho}} := \vec{s}]) = (\mathsf{FV}(r) \setminus \vec{x}^{\vec{\rho}}) \cup \bigcup\{\mathsf{FV}(s_i) | x_i^{\rho_i} \in \mathsf{FV}(r) \wedge x_i^{\rho_i} \neq x_j^{\rho_j} \text{ for } j < i\}$ as follows:

(V) $x^\rho[\vec{x}^{\vec{\rho}} := \vec{s}] := x^\rho$, if x^ρ does not occur in the list $\vec{x}^{\vec{\rho}}$.
$x^\rho[\vec{x}^{\vec{\rho}} := \vec{s}] := s_i$, if i is the smallest index such that $x_i^{\rho_i} = x^\rho$. Proof trivial.

($\to$-I) $(\lambda x^\rho r)[\vec{x}^{\vec{\rho}} := \vec{s}] := \lambda x^\rho.r[\vec{x}^{\vec{\rho}} := \vec{s}]$ where according to the variable convention and the infinite supply of untyped term variables we assume that x^ρ is different from the $\vec{x}^{\vec{\rho}}$ and does not occur free in any of the $\vec{s}$. The proof of the second claim uses the assumption.

($\to$-E) $(rs)[\vec{x}^{\vec{\rho}} := \vec{s}] := r[\vec{x}^{\vec{\rho}} := \vec{s}]s[\vec{x}^{\vec{\rho}} := \vec{s}]$. Proof obvious.

($\forall$-I) $(\Lambda\alpha r)[\vec{x}^{\vec{\rho}} := \vec{s}] := \Lambda\alpha.r[\vec{x}^{\vec{\rho}} := \vec{s}]$, where we assume that α does not occur free in any of the terms $\vec{s}$. $\Lambda\alpha.r[\vec{x}^{\vec{\rho}} := \vec{s}]$ is a term because of the assumption, Lemma 2.9, the second claim for r and the well-formedness of $\Lambda\alpha r$.

($\forall$-E) $(r\sigma)[\vec{x}^{\vec{\rho}} := \vec{s}] := r[\vec{x}^{\vec{\rho}} := \vec{s}]\sigma$. Proof obvious.

This definition is compatible with the variable convention.
The assumption in the rule ($\to$-I) will be written as $x^\rho \notin \vec{x}^{\vec{\rho}} \cup \mathsf{FV}(\vec{s})$ and gives us for $x \neq y$

- $(\lambda x^\alpha x^\alpha)[x^\alpha := y^\alpha] \neq \lambda x^\alpha y^\alpha$
- $(\lambda x^\alpha y^\alpha)[y^\alpha := x^\alpha] \neq \lambda x^\alpha x^\alpha$

The assumption in the rule ($\forall$-I) will be written as $\alpha \notin \mathsf{FTV}(\vec{s})$ and gives us for $\alpha \neq \beta$

- $(\Lambda\alpha x^\beta)[x^\beta := y^{\forall\alpha\beta}\alpha] \neq \Lambda\alpha.y^{\forall\alpha\beta}\alpha$
- $(\Lambda\alpha x^\beta)[x^\beta := y^{\alpha\to\beta}z^\alpha] \neq \Lambda\alpha.y^{\alpha\to\beta}z^\alpha$, where the object on the right is not even a term.

For ease of reference we state

Lemma 2.10 $\mathsf{FV}(r[\vec{x}^{\vec{\rho}} := \vec{s}]) = (\mathsf{FV}(r) \setminus \vec{x}^{\vec{\rho}}) \cup \bigcup\{\mathsf{FV}(s_i) | x_i^{\rho_i} \in \mathsf{FV}(r) \wedge x_i^{\rho_i} \neq x_j^{\rho_j} \text{ for } j < i\}$, which trivially implies $\mathsf{FV}(r[\vec{x}^{\vec{\rho}} := \vec{s}]) \subseteq (\mathsf{FV}(r) \setminus \vec{x}^{\vec{\rho}}) \cup \mathsf{FV}(\vec{s})$.

Proof Inside the definition. □

Lemma 2.11 If $\vec{x}^{\vec{\rho}} \cap \mathsf{FV}(r) = \emptyset$ then $r[\vec{x}^{\vec{\rho}} := \vec{s}] = r$.

Proof Induction on r. □

Lemma 2.12 $\mathsf{FTV}(r[\vec{x}^{\vec{\rho}} := \vec{s}]) = \mathsf{FTV}(r) \cup \bigcup\{\mathsf{FTV}(s_i) | x_i^{\rho_i} \in \mathsf{FV}(r) \wedge x_i^{\rho_i} \neq x_j^{\rho_j} \text{ for } j < i\}$

Proof Induction on r. The proof is easy but tedious. The case (V) makes use of Lemma 2.8. □

Lemma 2.13 If $x^\rho \notin \vec{x}^{\vec{\rho}} \cup \mathsf{FV}(\vec{s})$, then $(r[x^\rho := s])[\vec{x}^{\vec{\rho}} := \vec{s}] = (r[\vec{x}^{\vec{\rho}} := \vec{s}])[x^\rho := s[\vec{x}^{\vec{\rho}} := \vec{s}]]$.

Proof Induction on r. Note the similarity to Lemma 2.5. □

Lemma 2.14 If $x^\rho \notin \vec{x}^{\vec{\rho}} \cup \mathsf{FV}(\vec{s})$, then $r[\vec{x}^{\vec{\rho}}, x^\rho := \vec{s}, s] = (r[\vec{x}^{\vec{\rho}} := \vec{s}])[x^\rho := s]$.

Proof Induction on r. Note the similarity to Lemma 2.6. $\square$

Define the resulting term $r[\vec{\alpha} := \vec{\sigma}]$ of simultaneously substituting the finite list of types $\vec{\sigma}$ for the (equally long) list of type variables $\vec{\alpha}$ in r by recursion on r and simultaneously prove that $r^\rho[\vec{\alpha} := \vec{\sigma}] : \rho[\vec{\alpha} := \vec{\sigma}]$ and $\mathsf{FV}(r[\vec{\alpha} := \vec{\sigma}]) = \{y^{\tau[\vec{\alpha} := \vec{\sigma}]} | y^\tau \in \mathsf{FV}(r)\}$ as follows:

(V) $x^\rho[\vec{\alpha} := \vec{\sigma}] := x^{\rho[\vec{\alpha} := \vec{\sigma}]}$. Proof trivial.

($\to$-I) $(\lambda x^\rho r)[\vec{\alpha} := \vec{\sigma}] := \lambda x^{\rho[\vec{\alpha} := \vec{\sigma}]}.r[\vec{\alpha} := \vec{\sigma}]$, where we assume that for every $x^\sigma \in \mathsf{FV}(r)$ we have $\sigma = \rho$. Proof obvious for the first claim, the assumption is needed for the second claim when proving "$\supseteq$".

($\to$-E) $(rs)[\vec{\alpha} := \vec{\sigma}] := r[\vec{\alpha} := \vec{\sigma}]s[\vec{\alpha} := \vec{\sigma}]$. Proof obvious.

($\forall$-I) $(\Lambda\alpha r)[\vec{\alpha} := \vec{\sigma}] := \Lambda\alpha.r[\vec{\alpha} := \vec{\sigma}]$, where we assume that α is different from the $\vec{\alpha}$ and does not occur free in any of the $\vec{\sigma}$ (as in the definition of $(\forall\alpha\rho)[\vec{\alpha} := \vec{\sigma}]$). The proof of the first claim rests on this assumption, the other is obvious. $\Lambda\alpha.r[\vec{\alpha} := \vec{\sigma}]$ is a term because of this assumption, the second claim for r, Lemma 2.3 and the well-formedness of $\Lambda\alpha r$.

($\forall$-E) $(r\sigma)[\vec{\alpha} := \vec{\sigma}] := r[\vec{\alpha} := \vec{\sigma}]\sigma[\vec{\alpha} := \vec{\sigma}]$. The proof of the first claim is easy by using Lemma 2.5, the other is obvious.

This definition is again compatible with the variable convention.
Due to the assumption in the rule ($\to$-I) we have for $\alpha \neq \beta$ that $(\lambda x^\alpha x^\beta)[\beta := \alpha] \neq \lambda x^\alpha x^\alpha$. The assumption in ($\forall$-I) gives for $\alpha \neq \beta$

- $(\Lambda\alpha\lambda x^\alpha x^\alpha)[\alpha := \beta] \neq \Lambda\alpha\lambda x^\beta x^\beta$
- $(\Lambda\alpha x^\beta)[\beta := \alpha] \neq \Lambda\alpha x^\alpha$, where the object to the right is not even a term.

For ease of reference we state

Lemma 2.15 $\mathsf{FV}(r[\vec{\alpha} := \vec{\sigma}]) = \{y^{\tau[\vec{\alpha} := \vec{\sigma}]} | y^\tau \in \mathsf{FV}(r)\}$.

Proof Inside the definition. $\square$

Let "$x \notin \mathsf{FV}(r)$" be an abbreviation for "there is no x^ρ in $\mathsf{FV}(r)$". Hence, the lemma implies: $x \notin \mathsf{FV}(r) \Rightarrow x \notin \mathsf{FV}(r[\vec{\alpha} := \vec{\sigma}])$.

Lemma 2.16 If $\vec{\alpha} \cap \mathsf{FTV}(r) = \emptyset$ then $r[\vec{\alpha} := \vec{\sigma}] = r$.

Proof Induction on r. $\square$

Lemma 2.17 $\mathsf{FTV}(r[\vec{\alpha} := \vec{\sigma}]) = (\mathsf{FTV}(r) \setminus \vec{\alpha}) \cup \bigcup\{\mathsf{FV}(\sigma_i) | \alpha_i \in \mathsf{FTV}(r) \wedge \alpha_i \neq \alpha_j \text{ for } j < i\}$.

Proof Induction on r. The proof is tedious but routine. The case (V) makes use of Lemma 2.3. $\square$

Lemma 2.18 If $\alpha \notin \vec{\alpha} \cup \mathsf{FV}(\vec{\sigma})$, then $(r[\alpha := \sigma])[\vec{\alpha} := \vec{\sigma}] = (r[\vec{\alpha} := \vec{\sigma}])[\alpha := \sigma[\vec{\alpha} := \vec{\sigma}]]$.

Proof Induction on r. The proof makes use of Lemma 2.5 several times. $\square$

Lemma 2.19 If $\alpha \notin \vec{\alpha} \cup \mathsf{FV}(\vec{\sigma})$, then $r[\vec{\alpha}, \alpha := \vec{\sigma}, \sigma] = (r[\vec{\alpha} := \vec{\sigma}])[\alpha := \sigma]$.

Proof Induction on r. The proof makes use of Lemma 2.6 several times. $\square$

The following lemma allows to interchange type substitution and term substitution. The formulation is as general as possible. The only cases which are needed in the sequel are encapsulated in the two corollaries to the lemma. Readers who are not so much interested in substitution should simply check whether they can accept the statements made in the corollaries.

Lemma 2.20 (Interchange law for substitution) If for every $x^\rho \in \mathsf{FV}(r)$ and every i such that i is the minimal index with $x^{\rho[\vec{\alpha} := \vec{\sigma}]} = x_i^{\rho_i[\vec{\alpha} := \vec{\sigma}]}$, $\rho = \rho_i$ holds, then

$$r[\vec{x}^{\vec{\rho}} := \vec{s}][\vec{\alpha} := \vec{\sigma}] = r[\vec{\alpha} := \vec{\sigma}][\vec{x}^{\vec{\rho}[\vec{\alpha} := \vec{\sigma}]} := \vec{s}[\vec{\alpha} := \vec{\sigma}]].$$

Proof Induction on r. (V) Case x^ρ. If $x^\rho \in \vec{x}^{\vec{\rho}}$, let i be the minimal index with $x^\rho = x_i^{\rho_i}$. Then $x^\rho[\vec{x}^{\vec{\rho}} := \vec{s}][\vec{\alpha} := \vec{\sigma}] = s_i[\vec{\alpha} := \vec{\sigma}]$. Due to the assumption on free variables, i is also the minimal index such that $x^{\rho[\vec{\alpha} := \vec{\sigma}]} = x_i^{\rho_i[\vec{\alpha} := \vec{\sigma}]}$. Therefore $x^\rho[\vec{\alpha} := \vec{\sigma}][\vec{x}^{\vec{\rho}[\vec{\alpha} := \vec{\sigma}]} := \vec{s}[\vec{\alpha} := \vec{\sigma}]] = s_i[\vec{\alpha} := \vec{\sigma}]$. If $x^\rho \notin \vec{x}^{\vec{\rho}}$, then due to the assumption, $x^{\rho[\vec{\alpha} := \vec{\sigma}]} \notin \vec{x}^{\vec{\rho}[\vec{\alpha} := \vec{\sigma}]}$. The claim then follows from Lemma 2.11.
($\to$-I) Case $\lambda x^\rho r$. We may assume $x^{\rho[\vec{\alpha} := \vec{\sigma}]} \notin \vec{x}^{\vec{\rho}[\vec{\alpha} := \vec{\sigma}]} \cup \mathsf{FV}(\vec{s}[\vec{\alpha} := \vec{\sigma}])$ and that the following implication holds: $x^\sigma \in \mathsf{FV}(r) \cup \mathsf{FV}(\vec{s}) \Rightarrow \sigma = \rho$. By using Lemma 2.15 and Lemma 2.10 we get

$$(\lambda x^\rho r)[\vec{x}^{\vec{\rho}} := \vec{s}][\vec{\alpha} := \vec{\sigma}] = \lambda x^{\rho[\vec{\alpha} := \vec{\sigma}]}.r[\vec{x}^{\vec{\rho}} := \vec{s}][\vec{\alpha} := \vec{\sigma}] \quad \text{and}$$

$$(\lambda x^\rho r)[\vec{\alpha} := \vec{\sigma}][\vec{x}^{\vec{\rho}[\vec{\alpha} := \vec{\sigma}]} := \vec{s}[\vec{\alpha} := \vec{\sigma}]] = \lambda x^{\rho[\vec{\alpha} := \vec{\sigma}]}.r[\vec{\alpha} := \vec{\sigma}][\vec{x}^{\vec{\rho}[\vec{\alpha} := \vec{\sigma}]} := \vec{s}[\vec{\alpha} := \vec{\sigma}]].$$

Due to $x^{\rho[\vec{\alpha} := \vec{\sigma}]} \notin \vec{x}^{\vec{\rho}[\vec{\alpha} := \vec{\sigma}]}$, the induction hypothesis applies to r.
($\to$-E) Case rs. Use the induction hypothesis.
($\forall$-I) Case $\Lambda\alpha r$. We may assume that $\alpha \notin \vec{\alpha} \cup \mathsf{FV}(\vec{\sigma}) \cup \mathsf{FTV}(\vec{s})$. Using Lemma 2.17 we get

$$(\Lambda\alpha r)[\vec{x}^{\vec{\rho}} := \vec{s}][\vec{\alpha} := \vec{\sigma}] = \Lambda\alpha.r[\vec{x}^{\vec{\rho}} := \vec{s}][\vec{\alpha} := \vec{\sigma}] \quad \text{and}$$

$$(\Lambda\alpha r)[\vec{\alpha} := \vec{\sigma}][\vec{x}^{\vec{\rho}[\vec{\alpha} := \vec{\sigma}]} := \vec{s}[\vec{\alpha} := \vec{\sigma}]] = \Lambda\alpha.r[\vec{\alpha} := \vec{\sigma}][\vec{x}^{\vec{\rho}[\vec{\alpha} := \vec{\sigma}]} := \vec{s}[\vec{\alpha} := \vec{\sigma}]].$$

The claim follows by the induction hypothesis.
($\forall$-E) Case $r\sigma$. Use the induction hypothesis. $\square$

The condition in the lemma would be trivially satisfied if $\vec{x}^{\vec{\rho}} \cup \mathsf{FV}(r)$ were compatible in the sense that if $x^\rho, x^\sigma \in \vec{x}^{\vec{\rho}} \cup \mathsf{FV}(r)$ then $\rho = \sigma$. Call such a term r regular. Obviously, regularity is not closed under application of terms. If one wanted to enforce regularity everwhere application would have to be guarded by some compatibility requirement which would be nothing else than assuming the existence of a common variable context of $\vec{x}$ and the terms put together via application. Instead of having implicit contexts, one could then go one step further and have a type assignment system which in my view would give a one-sided sequent calculus instead of natural deduction[3].

Corollary 2.21 If $\vec{\alpha} \cap \mathsf{FV}(\vec{\rho}) = \emptyset$ and for every $x^\rho \in \mathsf{FV}(r)$, $\vec{\alpha} \cap \mathsf{FV}(\rho) = \emptyset$, then

$$r[\vec{x}^{\vec{\rho}} := \vec{s}][\vec{\alpha} := \vec{\sigma}] = r[\vec{\alpha} := \vec{\sigma}][\vec{x}^{\vec{\rho}} := \vec{s}[\vec{\alpha} := \vec{\sigma}]].$$

Proof Immediate from Corollary 2.2. $\square$

[3] The reader may decide whether is it worth having unrestricted application in the later chapters on inductive types at the price of unintuitive conditions in lemmas on substitution.

Corollary 2.22 If for every σ, $x^\sigma \in \mathsf{FV}(r)$ implies $\sigma = \rho$, then

$$r[x^\rho := s][\vec{\alpha} := \vec{\sigma}] = r[\vec{\alpha} := \vec{\sigma}][x^{\rho[\vec{\alpha}:=\vec{\sigma}]} := s[\vec{\alpha} := \vec{\sigma}]].$$

Proof Immediate from Lemma 2.20. □

Terms r with $\mathsf{FV}(r) = \emptyset$ are called closed terms. They may contain free type variables, i. e., $\mathsf{FTV}(r) \neq \emptyset$ is allowed for closed terms. Terms without free type variables have no special name. From now on we will use the names "variable convention" and "renaming convention" as synonyms.
An immediate subterm of a term r is any term r' which is explicitly mentioned as a term (by requiring $r' \in \Lambda$) in the definition clause by which the term r is introduced in Λ, i. e., x^ρ has no immediate subterm, $\lambda x^\rho r$ has only the immediate subterm r, rs has exactly the immediate subterms r and s, $\Lambda\alpha r$ has only the immediate subterm r and $r\sigma$ also has only the immediate subterm r. The relation "is a subterm of" is defined as the transitive reflexive closure of "is an immediate subterm of". As expected, the relation "is a proper subterm of" is defined as the transitive closure of "is an immediate subterm of".
The Curry-Howard isomorphism means on the term level that we see the term variables as symbols for proof assumptions, the typed term variables as assumptions of formulas and the typed terms as intuitionistic natural deduction proofs of their types seen as formulas. Hence, the term system of F "is" a proof system for intuitionistic (even minimal) second-order propositional logic (based on universal quantification over propositions and implication). The eigenvariable condition of ($\forall$-I) is hence the standard condition of second-order propositional logic: In order to deduce $\forall\alpha\rho$ from ρ we have to know that no assumption in the proof of ρ is dependent on α in the sense that α occurs free in the assumed formula.

2.1.3 Head form and normal forms

For terms r and finite lists $\vec{s}$ of typed terms **and** types we define when $r\vec{s}$ is defined and in that case a term it denotes by recursion on the length of the list as follows: If the list $\vec{s}$ is empty, $r\vec{s}{:=}r$. Otherwise $\vec{s}$ decomposes into the list $\vec{s}_-$ and the last element of $\vec{s}$. $r\vec{s}$ is only defined if $r\vec{s}_-$ is a term with type $\rho \to \sigma$ or $\forall\alpha\rho$. In the first case $r\vec{s}$ is defined iff the last element of $\vec{s}$ is a term s of type ρ, and set to $(r\vec{s}_-)s$ of type σ. In the second case $r\vec{s}$ is defined iff the last element of $\vec{s}$ is a type σ, and set to $(r\vec{s}_-)\sigma$ of type $\rho[\alpha := \sigma]$.
In case $r\vec{s}$ is defined, it is nothing but $(\cdots(rs_1)\cdots s_n)$, assuming $\vec{s}$ has n elements.
It is trivial to see that this definition is compatible with the renaming convention.
Whenever $r\vec{s}$ will be used as a term it is understood that r is a term and $\vec{s}$ is a finite list of typed terms and types such that $r\vec{s}$ is defined.

Lemma 2.23 If $r\vec{s}$ is a term and r' is a term of the same type as r, $r'\vec{s}$ is a term of the same type as $r\vec{s}$.

Proof By induction on the definition. □

Lemma 2.24 (Head form) Every term has exactly one of the following forms:

(V) $x^\rho\vec{s}$ (Terms of this form are called neutral.)

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

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

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

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

Proof Induction on terms (unicity is obvious). Consider e. g. the case ($\to$-E): If $r^{\rho\to\sigma}$ has one of the forms (V), ($\to$-R) or ($\forall$-R), rs has the same form. Otherwise, by induction hypothesis and because r is of arrow type, r is of form ($\to$-I), hence rs of form ($\to$-R). □

Define the set of terms in normal form NF as a subset of Λ by induction as follows:

(V) If $x^\rho\vec{s}$ is a term and the terms in $\vec{s}$ are in NF, then $x^\rho\vec{s} \in \mathsf{NF}$.

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

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

Most of the work in this thesis is concerned with proving strong normalization of β-reduction. The β-reduction relation of the next section is specifically aimed at allowing to transform any term of system F to a term in NF. The preceding lemma shows exactly what is gained by this transformation.

Lemma 2.25 If $r \in \mathsf{NF}$ and $\vec{s}^{\vec{\rho}} \subset \mathsf{NF}$ and $\vec{s}$ are neutral, then $r[\vec{x}^{\vec{\rho}} := \vec{s}] \in \mathsf{NF}$.

Proof Induction on $r \in \mathsf{NF}$. □

Lemma 2.26 If $r \in \mathsf{NF}$, then $r[\vec{\alpha} := \vec{\sigma}] \in \mathsf{NF}$.

Proof Induction on $r \in \mathsf{NF}$. □

Lemma 2.27 If $r \in \mathsf{NF}$ and r' is a subterm of r, then $r' \in \mathsf{NF}$.

Proof By induction on $r \in \mathsf{NF}$ (never using the induction hypothesis) show the statement for immediate subterms. □

In order to facilitate extensions to the system we give an equivalent definition of $r\vec{s}$ as an inductive definition as follows:

($\emptyset$) $r\emptyset := r$.

($\to$-E) If $r\vec{s} : \rho \to \sigma$ and $s : \rho$, then $r(\vec{s}, s) := (r\vec{s})s$.

($\forall$-E) If $r\vec{s} : \forall\alpha\rho$, then $r(\vec{s}, \sigma) := (r\vec{s})\sigma$.

Of course, $\emptyset$ means the empty list and ($\vec{s}$, object) means the "consed" list.
The inductive definition seems to be more natural than the recursive one, and it is obviously equivalent. (The well-definedness rests on the fact that the rules are mutually exclusive.)
If $\vec{s} : \vec{\rho}$ and $r\vec{s} : \rho$ it is convenient to write the type of r as $\vec{\rho} \to \rho$. More formally, we (inductively) define $\emptyset \to \rho := \rho$ and $(\vec{\rho}, \rho) \to \sigma := \vec{\rho} \to (\rho \to \sigma)$.

2.1.4 Reduction

Define a binary relation $\mapsto$ on terms by giving the only cases where it holds as follows:

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

Of course, we assume that the objects in the rules are terms. It is an important property of the rules that if the object to the left is a term then the object to the right is also a term, and moreover that it has the same type.
If $r \mapsto r'$ we say that r reduces to r' by one outer application of one of the β reduction rules. The terms on the left are called β-redices, the terms on the right their contracta.
Define a binary relation $\to_{whd}$ on terms by the following rule: If $r \mapsto r'$ and $r\vec{s}$ is a term, then $r\vec{s} \to_{whd} r'\vec{s}$. The relation $\to_{whd}$ is called weak head reduction. Due to Lemma 2.23 $r'\vec{s}$ is also a term and of the same type as $r\vec{s}$. Notice that exactly those terms allow weak head reduction which are listed in rules with an "R" in the name in the lemma on the head form.
Inductively define a binary relation $\to$ on terms[4] (and prove that if $r \to r'$, then r and r' have the same type and $\mathsf{FV}(r') \subseteq \mathsf{FV}(r)$) by the following rules:

(β) If $r \mapsto r'$, then $r \to r'$.

($\to$-I) If $r \to r'$, then $\lambda x^\rho r \to \lambda x^\rho r'$.

($\to$-E) If $r \to r'$, then $rs \to r's$. If $s \to s'$, then $rs \to rs'$.

($\forall$-I) If $r \to r'$, then $\Lambda \alpha r \to \Lambda \alpha r'$.

($\forall$-E) If $r \to r'$, then $r\sigma \to r'\sigma$.

(The proof for the clause (β) uses Lemma 2.10, Lemma 2.15 and the eigenvariable condition. The other proofs are obvious.) If $r \to r'$ we say that r reduces to r' by one application of one of the β reduction rules. We also describe this situation by saying that r reduces to r' in one step. The fact that r' has type ρ if r has type ρ is often (especially in type assignment systems) called the subject reduction property. From the proof-theoretic perspective (via the Curry-Howard isomorphism) the reduction relation is completely justified by type-preservation and the fact that the set of free variables is (weakly) decreasing: The reduction transforms proofs of a fixed formula without introducing new assumptions and without changing the assumed formulas (more precisely, without changing the assumption formulas of the assumptions remaining in the proof). In the formulation above one should have been more careful in order to express that if $r \to r'$ the well-formedness of r' follows from the well-formedness of r. In the case ($\forall$-I) the additional claim $\mathsf{FV}(r') \subseteq \mathsf{FV}(r)$ is used to guarantee this implication. This gives a better operational reading to the reduction relation: We may actually calculate with terms by applying β-reduction and do not have to check well-formedness over and over again.
Routinely we observe that the relations $\mapsto$, $\to_{whd}$ and $\to$ are compatible with the renaming convention.

Lemma 2.28 If $r \to r'$, then $\mathsf{FTV}(r') \subseteq \mathsf{FTV}(r)$.

Proof Induction on the definition of $\to$. The proof for clause (β) needs Lemma 2.12 and Lemma 2.17. □

[4] This shall mean that every clause has the implicit condition that every expression is in fact a term. Therefore, in ($\to$-E) we do not explicitly require that the types of r and s fit together, in ($\forall$-I) we do not explicitly impose the proviso and in ($\forall$-E) we do not state that r is of universal type.

Define the reducing eliminations $\hat{\mathrm{E}}_\to$ and $\hat{\mathrm{E}}_\forall$ by setting

$$r^{\rho\to\sigma}\hat{\mathrm{E}}_\to s^\rho := \begin{cases} t[x^\rho := s] & \text{, if } r = \lambda x^\rho t^\sigma \\ rs & \text{, else}\end{cases}$$
$$r^{\forall\alpha\rho}\hat{\mathrm{E}}_\forall \sigma := \begin{cases} t[\alpha := \sigma] & \text{, if } r = \Lambda \alpha t^\rho \\ r\sigma & \text{, else}\end{cases}$$

Hence, $rs(= \cup \mapsto)r\hat{\mathrm{E}}_\to s$ and $r\sigma(= \cup \mapsto)r\hat{\mathrm{E}}_\forall\sigma$.

Lemma 2.29 If $r \in \mathsf{NF}$ and s neutral and $s \in \mathsf{NF}$ and $r\hat{\mathrm{E}}_\to s$ is defined, then $r\hat{\mathrm{E}}_\to s \in \mathsf{NF}$. If $r \in \mathsf{NF}$ and $r\hat{\mathrm{E}}_\forall\sigma$ is defined, then $r\hat{\mathrm{E}}_\forall\sigma \in \mathsf{NF}$.

Proof Use case distinction on $r \in \mathsf{NF}$ and Lemma 2.25 and Lemma 2.26. □

Define the relation $\to^*$ as the reflexive transitive closure of $\to$ and the relation $\to^+$ as the transitive closure of $\to$. If $r \to^* r'$ holds we say that r reduces in finitely many steps to r', and if $r \to^+ r'$ holds we say that r reduces in at least one step to r'.

Lemma 2.30 If $r \to^* r'$, then $\mathsf{FV}(r') \subseteq \mathsf{FV}(r)$, $\mathsf{FTV}(r') \subseteq \mathsf{FTV}(r)$ and r and r' have the same type.

Proof Trivial consequence of the respective statements for $\to$. □

Lemma 2.31 If $r \to r'$, then $r[\vec{x}^{\vec{\rho}} := \vec{s}] \to r'[\vec{x}^{\vec{\rho}} := \vec{s}]$ and $r[\vec{\alpha} := \vec{\sigma}] \to r'[\vec{\alpha} := \vec{\sigma}]$. If $s \to s'$, then $r[x^\rho := s] \to^* r[x^\rho := s']$ (the number of steps is the number of free occurrences of x^ρ in r).

Proof Induction on the definition of $\to$ for the first two statements using Corollary 2.21, Corollary 2.22, Lemma 2.13 and Lemma 2.18 (nearly everything we know on interchange of substitutions) for the initial cases of the definition of $\to$. The last statement is easily proved by induction on r. □

Define $\mathsf{nf} := \{r \in \Lambda | \forall r'. r \not\to r'\}$ as the set of normal terms with respect to $\to$. Define the set of weakly normalizing terms $\mathsf{wn} := \{r \in \Lambda | \exists r' \in \mathsf{nf}. r \to^* r'\}$.
Inductively define the set sn of strongly normalizing terms (with respect to $\to$) by the following clause:

If for every term r' with $r \to r'$ we have $r' \in \mathsf{sn}$, then $r \in \mathsf{sn}$.

This is a strictly positive inductive definition because in the premise of the clause a reference to sn is only made on the right side of a nonnested implication. The definition of sn even is an accessibility inductive definition [BFPS81, p. 22] and hence sn the accessible part of $\to$.

Lemma 2.32 sn is the set of terms r for which there is no infinite reduction sequence

$$r \to r_1 \to r_2 \to \dots$$

Proof $\subseteq$: By induction on $r \in \mathsf{sn}$ show that r has no infinite reduction sequence. Assume an infinite reduction sequence $r \to r_1 \to r_2 \to r_3 \to \dots$ This gives an infinite reduction sequence $r_1 \to r_2 \to r_3 \to \dots$ which is impossible by induction hypothesis.
$\supseteq$: This is an instance of the principle of bar induction [Fef77, p. 942] which says that wellfoundedness of a relation implies the principle of (transfinite) induction over this relation: Simply show that $\{r' | r \to^* r'\} \subseteq \mathsf{sn}$ by induction over $\to$ which is wellfounded on $\{r' | r \to^* r'\}$ by assumption. (The principle itself is very easily proved in classical set theory.) □

It will turn out that the use of the inductively defined set sn via induction makes proofs more elegant than those which directly refer to the non-existence of infinite reduction sequences[5]. These definitions will be used for any reduction relation $\to$ on terms. If every term of a system is strongly normalizing w.r.t. $\to$, then the system is called strongly normalizing w.r.t. $\to$. If $\to$ is the canonical reduction relation for that system then we say that the system is strongly normalizing. In this thesis $\to$ is always reserved for the reduction relation generated from β-reduction and every system which gets a name is strongly normalizing (cf. chapter 9).

Lemma 2.33 If $r \in \mathsf{nf}$ and r' is a subterm of r, then $r' \in \mathsf{nf}$.

Proof By induction on r show the statement for immediate subterms. □

Lemma 2.34 $\mathsf{nf} = \mathsf{NF}$, i. e., the normal terms are the terms in normal form.

Proof $\subseteq$: Induction on r using Lemma 2.33 and making case analysis according to the head form. $\supseteq$: Induction on $r \in \mathsf{NF}$. □

Define the set $\mathsf{WN} \subseteq \Lambda$ inductively as follows:

(V) If $x^\rho \vec{s}$ is a term and the terms in $\vec{s}$ are in WN, then $x^\rho \vec{s} \in \mathsf{WN}$.

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

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

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

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

Lemma 2.35 $\mathsf{WN} \subseteq \mathsf{wn}$.

Proof Induction on WN using Lemma 2.34. □

Note that due to the unicity statement in the lemma on the head form, the definition of WN is well-parsed in the sense, that any term can only enter WN by using a uniquely determined clause with uniquely determined ingredients earlier entered into WN. Therefore, we may define functions on WN by recursion on the definition.
Define the normalizing function Ω as a function from WN to Λ by recursion on WN and simultaneously prove that $r \to^* \Omega(r) \in \mathsf{NF}$ for $r \in \mathsf{WN}$ (thus ensuring type coincidence and compatibility with the variable convention) as follows:

(V) $\Omega(x^\rho \vec{s}) := x^\rho \Omega(\vec{s})$, where $\Omega(\vec{s})$ is the list of elements $\Omega(s_i)$ (setting $\Omega(\sigma) := \sigma$).

($\to$-I) $\Omega(\lambda x^\rho r) := \lambda x^\rho \Omega(r)$.

($\beta_\to$) $\Omega((\lambda x^\rho r) s \vec{s}) := \Omega(r[x^\rho := s]\vec{s})$.

($\forall$-I) $\Omega(\Lambda \alpha r) := \Lambda \alpha \Omega(r)$.

($\beta_\forall$) $\Omega((\Lambda \alpha r)\sigma \vec{s}) := \Omega(r[\alpha := \sigma]\vec{s})$.

[5] In [Alt93b] the use of sn is seen as "one of the main technical contributions which simplify the formalization of the proof" (p. 18) of strong normalization of system F. Note that it would be quite cumbersome to formalize the notion of infinite reduction sequences and the height of a reduction tree in a system of computer-assisted proof development such as LEGO (see the citations in [Alt93b]).

The proofs are always obvious. For reference purposes we formulate

Lemma 2.36 For every $r \in \mathsf{WN}$, $r \to^* \Omega(r) \in \mathsf{NF}$.

Proof Inside the definition. □

The function Ω may be seen as providing a "big step semantics" for the terms.

Lemma 2.37 There is no closed term of type $\forall\alpha\alpha$ in WN.

Proof Assume a closed term $r^{\forall\alpha\alpha} \in \mathsf{WN}$. Then $\Omega(r) \in \mathsf{NF}$, and $\Omega(r)$ is also a closed term of type $\forall\alpha\alpha$ (due to Lemma 2.30). It is immediate from the definition of NF that therefore $\Omega(r) = \Lambda\alpha s$ for a closed term $s^\alpha \in \mathsf{NF}$. Such a term s does not exist (again due to the definition of NF). □

Define the set $\mathsf{SN} \subseteq \Lambda$ inductively as follows:

(V) If $x^\rho \vec{s}$ is a term and the terms in $\vec{s}$ are in SN, then $x^\rho \vec{s} \in \mathsf{SN}$.

($\to$-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, $\boxed{s \in \mathsf{SN}}$ 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 $(\Lambda \alpha r)\sigma \vec{s} \in \mathsf{SN}$.

Lemma 2.38 $\mathsf{SN} \subseteq \mathsf{WN}$.

Proof Trivial induction on SN. □

Lemma 2.39 $\mathsf{sn} \subseteq \mathsf{wn}$.

Proof Induction on sn: Let $r \in \mathsf{sn}$. Assume the induction hypothesis that for all r' with $r \to r'$ we already have $r' \in \mathsf{wn}$. Either $r \in \mathsf{nf} \subseteq \mathsf{wn}$ or there is an r' with $r \to r'$. By assumption we have an $r'' \in \mathsf{nf}$ with $r' \to^* r''$, hence $r \to^* r'' \in \mathsf{nf}$ and therefore $r \in \mathsf{wn}$. (Note that this proof does not at all inspect the relation $\to$.) □

In chapter 9 we show that $\mathsf{SN} \subseteq \mathsf{sn}$ and that $\mathsf{SN} = \Lambda$, implying that $\mathsf{sn} = \mathsf{WN} = \mathsf{wn} = \Lambda$ and making the function Ω to a normalizing function for every term. (I like to call this naïve reduction-free normalization because the definition of Ω does not depend on the notion of reduction, but clearly models head reduction. Compare also [CV97].) Without this knowledge a standardization theorem (see e. g. [Bar84]) would be needed in order to prove $\mathsf{WN} = \mathsf{wn}$. We show that also without this knowledge we get $\mathsf{sn} \subseteq \mathsf{SN}$. We need a further definition. For every term $r \in \mathsf{sn}$ define the height $\nu(r) \in \mathbb{N}$ by induction on $r \in \mathsf{sn}$ as follows:

$$\nu(r) := \max\{1 + \nu(r') | r \to r'\},$$

where $\max \emptyset := 0$. This is well-defined because there are always only finitely many terms to which r reduces in one step.

Lemma 2.40 If $r \in \mathsf{sn}$ and r' is a subterm of r, we have $r' \in \mathsf{sn}$ and $\nu(r') \leq \nu(r)$.

Proof By induction on $r \in \mathsf{sn}$ show the statement for immediate subterms (doing case analysis according to the definition of immediate subterms). □

Lemma 2.41 $r \in \mathsf{sn}$ implies $r \in \mathsf{SN}$.

Proof Main induction on $\nu(r)$ and side induction on r employing case analysis according to the head form of r using Lemma 2.40 everywhere:
(V) Case $x^\rho \vec{s}$. For every term s_i, we have $\nu(s_i) \leq \nu(x^\rho \vec{s})$. By the side induction hypothesis, $s_i \in \mathsf{SN}$, hence $x^\rho \vec{s} \in \mathsf{SN}$.
($\to$-I) Case $\lambda x^\rho r$. $\nu(r) \leq \nu(\lambda x^\rho r)$. By the side induction hypothesis, $r \in \mathsf{SN}$, hence $\lambda x^\rho r \in \mathsf{SN}$.
($\to$-R) Case $(\lambda x^\rho r)s\vec{s}$. $\nu(r[x^\rho := s]\vec{s}) < \nu((\lambda x^\rho r)s\vec{s})$. By the main induction hypothesis, we get $r[x^\rho := s]\vec{s} \in \mathsf{SN}$. $\nu(s) \leq \nu((\lambda x^\rho r)s\vec{s})$. By the side induction hypothesis, $s \in \mathsf{SN}$. Altogether $(\lambda x^\rho r)s\vec{s} \in \mathsf{SN}$. ($\forall$-I) and ($\forall$-R) are somewhat easier to prove. □

It should be remarked that $\mathsf{sn} \subseteq \mathsf{SN}$ is true irrespective of types and hence could also be proved in a type-free version of the system. It will later (cf. chapter 9) turn out that the same is true for $\mathsf{SN} \subseteq \mathsf{sn}$. (Of course, one has to distinguish the $\forall$-elimination by some device as e. g. a constant symbol expressing any type.) It should also be remarked that ν is only used for ease of presentation. If $\to$ were infinitely branching a more careful analysis would also allow to prove the preceding lemma.
We conclude this section by giving a typical application of normalization, i. e., of the fact that $\mathsf{WN} = \Lambda$ to be proved in chapter 9. The proof of the following lemma does not use it, but its corollary does.

Lemma 2.42 If there is a closed term of type $\mathsf{peirce}(\rho)$[6] $:= ((\alpha \to \rho) \to \alpha) \to \alpha$ in NF, then there is a closed term of type $\alpha \to \rho$.

Proof Let $r^{\mathsf{peirce}(\rho)} \in \mathsf{NF}$ be closed. r is not of the form $x^\sigma \vec{s}$ because it is closed. Hence, $r = \lambda x^{(\alpha\to\rho)\to\alpha} s^\alpha$ with $s \in \mathsf{NF}$ and $\mathsf{FV}(s) \subseteq \{x^{(\alpha\to\rho)\to\alpha}\}$. Because α is not composed, $s = y^\sigma \vec{s}$. Because $\mathsf{FV}(s) \subseteq \{x^{(\alpha\to\rho)\to\alpha}\}$, $y^\sigma = x^{(\alpha\to\rho)\to\alpha}$ and hence $s = x^{(\alpha\to\rho)\to\alpha} t^{\alpha\to\rho}$ for $t \in \mathsf{NF}$ with $\mathsf{FV}(t) \subseteq \{x^{(\alpha\to\rho)\to\alpha}\}$. If $t = z^\tau \vec{t}$, then $z^\tau = x^{(\alpha\to\rho)\to\alpha}$ and hence $t = x^{(\alpha\to\rho)\to\alpha} t_1$ which is not type-correct. Therefore $t = \lambda z^\alpha u^\rho$ with $u \in \mathsf{NF}$ and $\mathsf{FV}(u) \subseteq \{x^{(\alpha\to\rho)\to\alpha}, z^\alpha\}$. Consider $u' := u[x^{(\alpha\to\rho)\to\alpha} := \lambda h^{\alpha\to\rho}.z^\alpha]$. $u' : \rho$ and $\mathsf{FV}(u') \subseteq \{z^\alpha\}$. Hence, $\lambda z^\alpha u'$ is a closed term of type $\alpha \to \rho$. □

Corollary 2.43 If $\Lambda = \mathsf{WN}$, then there is no closed term of type $\mathsf{peirce}(\beta)$ for $\beta \neq \alpha$.

Proof If there were a closed $r^{\mathsf{peirce}(\rho)} \in \Lambda = \mathsf{WN}$, then $\Omega(r) \in \mathsf{NF}$ closed and of the same type. By the preceding lemma, there would be a closed term $s^{\alpha\to\beta}$. Then by $\Lambda = \mathsf{WN}$, $(\Omega(s))^{\alpha\to\beta} \in \mathsf{NF}$ and closed. Therefore, $\Omega(s)$ cannot have the form $x^\rho \vec{s}$ and hence $\Omega(s) = \lambda x^\alpha t$ with $t^\beta \in \mathsf{NF}$ and $\mathsf{FV}(t) \subseteq \{x^\alpha\}$. Because the type of t is not composed, t has to be of the form $y^\rho \vec{t}$ and due to $\mathsf{FV}(t) \subseteq \{x^\alpha\}$, $y^\rho = x^\alpha$. But this is incompatible with $t : \beta \neq \alpha$. Contradiction. □

Acknowledgement: 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 [vRS95] and [Loa95]. The idea of using WN and Ω came from an attempt to "re-engineer" the typed operational semantics in [Gog94] for the case of simply-typed lambda calculus[7].

[6] The name stems from the Peirce formula $((p \to q) \to p) \to p$ in first-order propositional logic where p and q are atoms. Although classically true it is not provable in intuitionistic logic.

[7] In dependent type theory as studied in [Gog94] one cannot separate the definitions of WN and Ω which makes typed operational semantics much more interesting. The essence of typed operational semantics is also shown in [Gog95] for the case of simply-typed lambda calculus.

2.2 The system **eF** which extends **F** by the types 0, 1, pairs and sums

We introduce some standard type constructs although they may be encoded in F even preserving reduction (section 2.2.6). In later extensions this is not always possible (even not in PIT). Moreover, the encoding is essentially impredicative (not only does it use the universal quantifier but it also destroys strict positivity as defined in section 3.2). Finally, η-reduction would not be preserved by the embedding, and $+$ gives rise to additional rewrite rules such as permutative conversions[8] (called commuting conversions in [GLT89] and studied at length in [Pra71]; an alternative normalization proof is given in [Lei75] and yet another in [Joa97]).

2.2.1 Types

The type system of eF is defined by adding the following clauses to the inductive definition of Types for F:

(0) $0 \in \mathsf{Types}$.

(1) $1 \in \mathsf{Types}$.

($\times$) If $\rho \in \mathsf{Types}$ and $\sigma \in \mathsf{Types}$, then $\rho \times \sigma \in \mathsf{Types}$.

($+$) If $\rho \in \mathsf{Types}$ and $\sigma \in \mathsf{Types}$, then $\rho + \sigma \in \mathsf{Types}$.

The renaming convention will be followed as for system F as well as the dot notation. We assume that $\forall$ binds stronger than all the binary connectives and that $+$ and $\times$ bind stronger than $\to$. The definition of free variables of a type is extended in the expected way by the following clauses:

(0) $\mathsf{FV}(0) := \emptyset$.

(1) $\mathsf{FV}(1) := \emptyset$.

($\times$) $\mathsf{FV}(\rho \times \sigma) := \mathsf{FV}(\rho) \cup \mathsf{FV}(\sigma)$.

($+$) $\mathsf{FV}(\rho + \sigma) := \mathsf{FV}(\rho) \cup \mathsf{FV}(\sigma)$.

The definition remains compatible with the renaming convention.
The definition of $\rho[\vec{\alpha} := \vec{\sigma}]$ is extended by the following clauses (as expected):

(0) $0[\vec{\alpha} := \vec{\sigma}] := 0$.

(1) $1[\vec{\alpha} := \vec{\sigma}] := 1$.

($\times$) $(\rho \times \sigma)[\vec{\alpha} := \vec{\sigma}] := \rho[\vec{\alpha} := \vec{\sigma}] \times \sigma[\vec{\alpha} := \vec{\sigma}]$.

($+$) $(\rho + \sigma)[\vec{\alpha} := \vec{\sigma}] := \rho[\vec{\alpha} := \vec{\sigma}] + \sigma[\vec{\alpha} := \vec{\sigma}]$.

It is again compatible with the renaming convention.
We adopt the same naming convention.
It is very easy to see that the statements on types given for system F are also true for eF.

[8] It is possible to extend the normalization proofs as presented in this thesis to also cover permutative conversions in a modular fashion.

2.2.2 Terms

The definition of typed terms and their free variables for system F is extended by the following clauses:

(0-E) If $r \in \Lambda_0$, then $r\rho \in \Lambda_\rho$. $\mathsf{FV}(r\rho) := \mathsf{FV}(r)$.

(1-I) $\mathsf{IN1} \in \Lambda_1$. $\mathsf{FV}(\mathsf{IN1}) := \emptyset$.

($\times$-I) If $r \in \Lambda_\rho$ and $s \in \Lambda_\sigma$, then $\langle r, s\rangle \in \Lambda_{\rho\times\sigma}$. $\mathsf{FV}(\langle r, s\rangle) := \mathsf{FV}(r) \cup \mathsf{FV}(s)$.

($\times$-E) If $r \in \Lambda_{\rho\times\sigma}$, then $r\mathsf{L} \in \Lambda_\rho$ and $r\mathsf{R} \in \Lambda_\sigma$. $\mathsf{FV}(r\mathsf{L}) := \mathsf{FV}(r)$ and $\mathsf{FV}(r\mathsf{R}) := \mathsf{FV}(r)$.

(+-I) If $r \in \Lambda_\rho$, then $\mathsf{INL}_\sigma r \in \Lambda_{\rho+\sigma}$. $\mathsf{FV}(\mathsf{INL}_\sigma r) := \mathsf{FV}(r)$. If $r \in \Lambda_\sigma$, then $\mathsf{INR}_\rho r \in \Lambda_{\rho+\sigma}$. $\mathsf{FV}(\mathsf{INR}_\rho r) := \mathsf{FV}(r)$.

(+-E) If $r \in \Lambda_{\rho+\sigma}$, $s \in \Lambda_{\rho\to\tau}$ and $t \in \Lambda_{\sigma\to\tau}$, then $rst \in \Lambda_\tau$. $\mathsf{FV}(rst) := \mathsf{FV}(r) \cup \mathsf{FV}(s) \cup \mathsf{FV}(t)$.

We will adopt all the conventions which were introduced for system F.
If $r : \rho \times \sigma$, we say that r is of product type. If $r : \rho + \sigma$, we say that r is of sum type.
Notice that without typing information on the subterms we cannot see which elimination rule (a rule which has an "E" in the name) is used. Ambiguities would e. g. arise with $x^{\forall\alpha\alpha}\alpha$ and $x^0\alpha$ (both in Λ_α) or with $x^{(\alpha\to\gamma)\to(\beta\to\gamma)\to\gamma}y^{\alpha\to\gamma}z^{\beta\to\gamma}$ and $x^{\alpha+\beta}y^{\alpha\to\gamma}z^{\beta\to\gamma}$ (both in Λ_γ) if the type of x is not shown. This phenomenon becomes a problem when one wants to speak about a term of the form $r\rho$ or of the form rst. In definitions by recursion on the definition of terms the problem is avoided by stating the name of the rule. As there are other ways of using terms we introduce a notation for eliminations. The definition of terms may as for system F be rewritten as follows (again leaving out the simultaneous definition of FV for terms):

(V) If $x \in \mathsf{Vars}$, then $x^\rho \in \Lambda$.

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

($\to$-E) If $r^{\rho\to\sigma} \in \Lambda$ and $s^\rho \in \Lambda$, then $(rs)^\sigma \in \Lambda$; also written as $r\mathrm{E}_\to s$.

($\forall$-I) If $r^\rho \in \Lambda$, then $(\Lambda\alpha r)^{\forall\alpha\rho} \in \Lambda$, provided that $\alpha \notin \mathsf{FV}(\sigma)$ for any σ such that there is an x^σ in $\mathsf{FV}(r)$.

($\forall$-E) If $r^{\forall\alpha\rho} \in \Lambda$, then $(r\sigma)^{\rho[\alpha:=\sigma]} \in \Lambda$; also written as $r\mathrm{E}_\forall\sigma$.

(0-E) If $r^0 \in \Lambda$, then $(r\rho)^\rho \in \Lambda$; also written as $r\mathrm{E}_0\rho$.

(1-I) $\mathsf{IN1}^1 \in \Lambda$.

($\times$-I) If $r^\rho \in \Lambda$ and $s^\sigma \in \Lambda$, then $\langle r, s\rangle^{\rho\times\sigma} \in \Lambda$.

($\times$-E) If $r^{\rho\times\sigma} \in \Lambda$, then $(r\mathsf{L})^\rho \in \Lambda$ and $(r\mathsf{R})^\sigma \in \Lambda$; also written as $r\mathrm{E}_\times\mathsf{L}$ and $\mathrm{E}_\times\mathsf{R}$.

(+-I) If $r^\rho \in \Lambda$, then $(\mathsf{INL}_\sigma r)^{\rho+\sigma} \in \Lambda$. If $r^\sigma \in \Lambda$, then $(\mathsf{INR}_\rho r)^{\rho+\sigma} \in \Lambda$.

(+-E) If $r^{\rho+\sigma} \in \Lambda$, $s^{\rho\to\tau} \in \Lambda$ and $t^{\sigma\to\tau} \in \Lambda$, then $(rst)^\tau \in \Lambda$; also written as $r\mathrm{E}_+\tau st$ or shorter $r\mathrm{E}_+st$.

Extend the recursive definition of $\mathsf{FTV}(r)$ by the following clauses (as expected):

(0-E) $\mathsf{FTV}(r\rho) := \mathsf{FTV}(r) \cup \mathsf{FV}(\rho)$.

(1-I) $\mathsf{FTV}(\mathsf{IN1}) := \emptyset$.

($\times$-I) $\mathsf{FTV}(\langle r, s\rangle) := \mathsf{FTV}(r) \cup \mathsf{FTV}(s)$.

($\times$-E) $\mathsf{FTV}(r\mathsf{L}) := \mathsf{FTV}(r)$. $\mathsf{FTV}(r\mathsf{R}) := \mathsf{FTV}(r)$.

(+-I) $\mathsf{FTV}(\mathsf{INL}_\sigma r) := \mathsf{FV}(\sigma) \cup \mathsf{FTV}(r)$. $\mathsf{FTV}(\mathsf{INR}_\rho r) := \mathsf{FV}(\rho) \cup \mathsf{FTV}(r)$.

(+-E) $\mathsf{FTV}(rst) := \mathsf{FTV}(r) \cup \mathsf{FTV}(s) \cup \mathsf{FTV}(t)$.

The definition of $r[\vec{x}^{\vec{\rho}} := \vec{s}]$ together with the proof of $r^\rho[\vec{x}^{\vec{\rho}} := \vec{s}] : \rho$ and $\mathsf{FV}(r[\vec{x}^{\vec{\rho}} := \vec{s}]) = (\mathsf{FV}(r) \setminus \vec{x}^{\vec{\rho}}) \cup \bigcup\{\mathsf{FV}(s_i) | x_i^{\rho_i} \in \mathsf{FV}(r) \wedge x_i^{\rho_i} \neq x_j^{\rho_j} \text{ for } j < i\}$ (i. e., the statement of Lemma 2.10) is extended by the following clauses:

(0-E) $(r\rho)[\vec{x}^{\vec{\rho}} := \vec{s}] := r[\vec{x}^{\vec{\rho}} := \vec{s}]\mathrm{E}_0\rho$. Proof obvious.

(1-I) $\mathsf{IN1}[\vec{x}^{\vec{\rho}} := \vec{s}] := \mathsf{IN1}$. Proof trivial.

($\times$-I) $\langle r, s\rangle[\vec{x}^{\vec{\rho}} := \vec{s}] := \langle r[\vec{x}^{\vec{\rho}} := \vec{s}], s[\vec{x}^{\vec{\rho}} := \vec{s}]\rangle$. Proof obvious.

($\times$-E) $(r\mathsf{L})[\vec{x}^{\vec{\rho}} := \vec{s}] := r[\vec{x}^{\vec{\rho}} := \vec{s}]\mathsf{L}$, $(r\mathsf{R})[\vec{x}^{\vec{\rho}} := \vec{s}] := r[\vec{x}^{\vec{\rho}} := \vec{s}]\mathsf{R}$. Proof obvious.

(+-I) $(\mathsf{INL}_\sigma r)[\vec{x}^{\vec{\rho}} := \vec{s}] := \mathsf{INL}_\sigma r[\vec{x}^{\vec{\rho}} := \vec{s}]$, $(\mathsf{INR}_\rho r)[\vec{x}^{\vec{\rho}} := \vec{s}] := \mathsf{INR}_\rho r[\vec{x}^{\vec{\rho}} := \vec{s}]$. Proof obvious.

(+-E) $(rst)[\vec{x}^{\vec{\rho}} := \vec{s}] := r[\vec{x}^{\vec{\rho}} := \vec{s}]\mathrm{E}_+s[\vec{x}^{\vec{\rho}} := \vec{s}]t[\vec{x}^{\vec{\rho}} := \vec{s}]$. Proof obvious.

Note that the new rules are in itself unproblematic but that the above statements have to be proved simultaneously in order to deal with the rules of system F not shown again.
The definition of $r[\vec{\alpha} := \vec{\sigma}]$ together with the proof of $r^\rho[\vec{\alpha} := \vec{\sigma}] : \rho[\vec{\alpha} := \vec{\sigma}]$ and $\mathsf{FV}(r[\vec{\alpha} := \vec{\sigma}]) = \{y^{\tau[\vec{\alpha}:=\vec{\sigma}]} | y^\tau \in \mathsf{FV}(r)\}$ is extended by the following clauses:

(0-E) $(r\rho)[\vec{\alpha} := \vec{\sigma}] := r[\vec{\alpha} := \vec{\sigma}]\mathrm{E}_0\rho[\vec{\alpha} := \vec{\sigma}]$. Proof obvious.

(1-I) $\mathsf{IN1}[\vec{\alpha} := \vec{\sigma}] := \mathsf{IN1}$. Proof trivial.

($\times$-I) $\langle r, s\rangle[\vec{\alpha} := \vec{\sigma}] := \langle r[\vec{\alpha} := \vec{\sigma}], s[\vec{\alpha} := \vec{\sigma}]\rangle$. Proof obvious.

($\times$-E) $(r\mathsf{L})[\vec{\alpha} := \vec{\sigma}] := r[\vec{\alpha} := \vec{\sigma}]\mathsf{L}$, $(r\mathsf{R})[\vec{\alpha} := \vec{\sigma}] := r[\vec{\alpha} := \vec{\sigma}]\mathsf{R}$. Proof obvious.

(+-I) $(\mathsf{INL}_\sigma r)[\vec{\alpha} := \vec{\sigma}] := \mathsf{INL}_{\sigma[\vec{\alpha}:=\vec{\sigma}]}r[\vec{\alpha} := \vec{\sigma}]$, $(\mathsf{INR}_\rho r)[\vec{\alpha} := \vec{\sigma}] := \mathsf{INR}_{\rho[\vec{\alpha}:=\vec{\sigma}]}r[\vec{\alpha} := \vec{\sigma}]$. Proof obvious.

(+-E) $(rst)[\vec{\alpha} := \vec{\sigma}] := r[\vec{\alpha} := \vec{\sigma}]\mathrm{E}_+s[\vec{\alpha} := \vec{\sigma}]t[\vec{\alpha} := \vec{\sigma}]$. Proof obvious.

Note that the families of term-forming constructs $(\mathsf{INL}_\rho)_\rho$ and $(\mathsf{INR}_\rho)_\rho$ are polymorphic in the sense that type substitution on terms also affects those indices.
The statements made in section 2.1.2 are valid in eF, too.
The general definition of immediate subterms of a term gives the following extension for system eF: $\mathsf{IN1}$ has no immediate subterm, $rE_0\rho$, $r\mathsf{L}$, $r\mathsf{R}$, $\mathsf{INL}_\sigma r$ and $\mathsf{INR}_\rho r$ have only the immediate subterm r, $\langle r, s\rangle$ has exactly the immediate subterms s and t and $r\mathrm{E}_+st$ has exactly the immediate subterms r, s and t.
Let us interpret eF via the Curry-Howard isomorphism: The type 0 represents $\bot$, 1 stands for $\top$, $\times$ for $\wedge$ and $+$ for $\vee$. The term formation rules are all valid in intuitionistic second-order propositional logic and give a natural deduction proof system. The reading as a proof system hopefully makes it easier to accept my decision to write eliminations as application: The main premise is written to the left, the side premisses are written to the right. And the terms are simply a linear notation for the deduction trees.

2.2.3 Head form and normal forms

For terms r and finite lists $\vec{s}$ of typed terms and types and symbols L and R we define when $r\vec{s}$ is defined and in that case a term it denotes by adding the following clauses to the inductive definition given for system F:

(0-E) If $r\vec{s} : 0$, then $r(\vec{s}, \rho) := (r\vec{s})\rho$.

($\times$-E) If $r\vec{s} : \rho \times \sigma$, then $r(\vec{s}, \mathsf{L}) := (r\vec{s})\mathsf{L}$ and $r(\vec{s}, \mathsf{R}) := (r\vec{s})\mathsf{R}$.

(+-E) If $r\vec{s} : \rho + \sigma$, $s : \rho \to \tau$ and $t : \sigma \to \tau$, then $r(\vec{s}, s, t) := (r\vec{s})st$.

We will use the same convention for $r\vec{s}$ as before.
The statement of Lemma 2.23 hold also true for eF. Lemma 2.24 has to be extended to:

Lemma 2.44 (Head form) Every term has exactly one of the following forms:

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

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

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

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

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

(1-I) $\mathsf{IN1}$

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

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

(+-I) $\mathsf{INL}_\rho r$, $\mathsf{INR}_\rho r$

(+-R) $\mathsf{INL}_\rho rst\vec{s}$, $\mathsf{INR}_\rho rst\vec{s}$

Proof Induction on terms (unicity again being obvious). Consider e. g. the case ($\to$-E): If $r^{\rho\to\sigma}$ has one of the forms (V), ($\to$-R), ($\forall$-R), ($\times$-R) or (+-R), rs has the same form. Otherwise, by induction hypothesis and because r is of arrow type, r is of form ($\to$-I), hence rs of form ($\to$-R). □

Extend the inductive definition of the set NF for system F by the following clauses:

(1-I) $\mathsf{IN1} \in \mathsf{NF}$.

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

(+-I) If $r \in \mathsf{NF}$, then $\mathsf{INL}_\rho r \in \mathsf{NF}$ and $\mathsf{INR}_\rho r \in \mathsf{NF}$.

The statements made in section 2.1.3 are valid in eF, too.
A general remark concerning the proofs of statements which are interpreted in an extended system is in order. If e. g. the statement of Lemma 2.25 is proved for system eF also by induction on $r \in \mathsf{NF}$, one has to consider all the defining clauses for NF in eF. The clauses for NF of system F appear verbatim in the definition of NF for eF, but they have a different interpretation simply because their variables are ranging over a larger domain. This does no harm to the validity of the respective clauses in the proof as they also can simply be reinterpreted as ranging over

this larger domain. But this reinterpretation is not always possible, e. g. if in the original proof clauses a case analysis on the terms is carried out. Then a "quadratic effect" on the proof urges us to extend the proof clauses to the new system and add completely new clauses. In my presentation there is hardly any such "quadratic reasoning" because embedded analyses are normally carried out beforehand and encapsulated in a lemma which has to be extended to the larger system before extending the lemma in consideration.
Lemma 2.44 shows a different phenomenon. The statement of Lemma 2.24 had to be extended to cover eF. Its proof by induction on terms has the additional clauses of the extended term definition, and the proof clauses corresponding to the term definition of system F have to be redone in order to capture the new statement. But as was shown in the proof in the case of ($\to$-E), this extension did not need any new arguments. If one used the expression "of any of the forms with an R in the name" the original proof clause would not have needed any change. Therefore, I consider this extension not to be "quadratic" in the following

Meta-Convention on Extensions: If a proof of a statement in the extension of a system is not explicitly given and the statement is the same as that for the original system or an extension of a statement in the original system, only "non-quadratic reasoning" will be needed for the proof. (This is nothing but a promise by the author clarifying the words "analogous" and "similar" which are thus avoided for system extensions.)

2.2.4 Reduction

Define the binary relation $\mapsto$ on terms by giving the only cases where it holds as follows:

$$\begin{array}{lrl}
(\beta_\to) & (\lambda x^\rho r)s & \mapsto r[x^\rho := s] \\
(\beta_\forall) & (\Lambda\alpha r)\sigma & \mapsto r[\alpha := \sigma] \\
(\beta_\times) & \langle r, s\rangle\mathsf{L} & \mapsto r \\
 & \langle r, s\rangle\mathsf{R} & \mapsto s \\
(\beta_+) & \mathsf{INL}_\rho rst & \mapsto sr \\
 & \mathsf{INR}_\rho rst & \mapsto tr
\end{array}$$

As for system F the objects on the right side are automatically terms if the respective object on the left is a term, and the types are equal.
The relation $\to_{whd}$ is defined as for system F and has the same properties as stated there.
Extend the inductive definition of the relation $\to$ for system F by the following clauses (and prove that if $r \to r'$, then r and r' have the same type and $\mathsf{FV}(r') \subseteq \mathsf{FV}(r)$):

(0-E) If $r \to r'$, then $r\rho \to r'\rho$.

($\times$-I) If $r \to r'$, then $\langle r, s\rangle \to \langle r', s\rangle$. If $s \to s'$, then $\langle r, s\rangle \to \langle r, s'\rangle$.

($\times$-E) If $r \to r'$, then $r\mathsf{L} \to r'\mathsf{L}$ and $r\mathsf{R} \to r'\mathsf{R}$.

(+-I) If $r \to r'$, then $\mathsf{INL}_\rho r \to \mathsf{INL}_\rho r'$ and $\mathsf{INR}_\rho r \to \mathsf{INR}_\rho r'$.

(+-E) If $r \to r'$, then $rst \to r'st$. If $s \to s'$, then $rst \to rs't$. If $t \to t'$, then $rst \to rst'$.

(The proofs were trivial. See also footnote 4 concerning well-formedness.)
In the light of the Curry-Howard isomorphism the reduction rules are again completely justified by the two properties which we just proved simultaneously with the definition: They are valid proof transformation rules.

Define the reducing eliminations $\hat{\mathrm{E}}_{\to}$ and $\hat{\mathrm{E}}_{\forall}$ as before and set

$$r^{\rho\times\sigma}\hat{\mathrm{E}}_{\times}\mathsf{L} \;:=\; \begin{cases} s & \text{, if } r = \langle s^{\rho}, t^{\sigma}\rangle \\ r\mathrm{E}_{\times}\mathsf{L} & \text{, else} \end{cases}$$

$$r^{\rho\times\sigma}\hat{\mathrm{E}}_{\times}\mathsf{R} \;:=\; \begin{cases} t & \text{, if } r = \langle s^{\rho}, t^{\sigma}\rangle \\ r\mathrm{E}_{\times}\mathsf{R} & \text{, else} \end{cases}$$

Hence, $r\mathrm{E}_{\times}\mathsf{L}(= \cup \mapsto)r\hat{\mathrm{E}}_{\times}\mathsf{L}$ and $r\mathrm{E}_{\times}\mathsf{R}(= \cup \mapsto)r\hat{\mathrm{E}}_{\times}\mathsf{R}$.
The statement of Lemma 2.29 is true for eF.

Lemma 2.45 If $r^{\rho\times\sigma} \in \mathsf{NF}$, then $r\hat{\mathrm{E}}_{\times}\mathsf{L} \in \mathsf{NF}$ and $r\hat{\mathrm{E}}_{\times}\mathsf{R} \in \mathsf{NF}$.

Proof Use case distinction on $r \in \mathsf{NF}$. □

Define the reducing $+$-elimination $\hat{\mathrm{E}}_{+}$ by

$$r^{\rho+\sigma}\hat{\mathrm{E}}_{+}st \;:=\; \begin{cases} sr' & \text{, if } r = \mathsf{INL}_{\sigma}r' \\ tr' & \text{, if } r = \mathsf{INR}_{\rho}r' \\ r\mathrm{E}_{+}st & \text{, else} \end{cases}$$

Obviously, this reducing elimination does not preserve normal forms because s and t may be λ-abstractions[9]. It will be used in section 8.2 for the confluence proof.
Extend the inductive definition of WN for system F by the following clauses:

(1-I) $\mathsf{IN1} \in \mathsf{WN}$.

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

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

($+$-I) If $r \in \mathsf{WN}$, then $\mathsf{INL}_{\rho}r \in \mathsf{WN}$ and $\mathsf{INR}_{\rho}r \in \mathsf{WN}$.

(β_{+}) If $(\mathsf{INL}_{\rho}r)st\vec{s}$ is a term and $sr\vec{s} \in \mathsf{WN}$, then $(\mathsf{INL}_{\rho}r)st\vec{s} \in \mathsf{WN}$. If $(\mathsf{INR}_{\rho}r)st\vec{s}$ is a term and $tr\vec{s} \in \mathsf{WN}$, then $(\mathsf{INR}_{\rho}r)st\vec{s} \in \mathsf{WN}$.

WN is again well-parsed and therefore definitions by recursion on WN are possible. Extend the recursive definition of the function Ω from WN to Λ and the simultaneous proof of $r \to^{*} \Omega(r) \in \mathsf{NF}$ for $r \in \mathsf{WN}$ by the following clauses:

(1-I) $\Omega(\mathsf{IN1}) := \mathsf{IN1}$.

($\times$-I) $\Omega(\langle r, s\rangle) := \langle\Omega(r), \Omega(s)\rangle$.

($\beta_{\times}$) $\Omega(\langle r, s\rangle\mathsf{L}\vec{s}) := \Omega(r\vec{s})$. $\Omega(\langle r, s\rangle\mathsf{R}\vec{s}) := \Omega(s\vec{s})$.

($+$-I) $\Omega(\mathsf{INL}_{\rho}r) := \mathsf{INL}_{\rho}\Omega(r)$ and $\Omega(\mathsf{INR}_{\rho}r) := \mathsf{INR}_{\rho}\Omega(r)$.

(β_{+}) $\Omega((\mathsf{INL}_{\rho}r)st\vec{s}) := \Omega(sr\vec{s})$. $\Omega((\mathsf{INR}_{\rho}r)st\vec{s}) := \Omega(tr\vec{s})$.

The proofs are always obvious.
Extend the inductive definition of SN for system F by the following clauses:

(1-I) $\mathsf{IN1} \in \mathsf{SN}$.

[9] One could optimize it but unless one studies additional conversions for sum types such as the variable elimination reduction sketched in appendix B this additional effort is not needed.

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

($+$-I) If $r \in \mathsf{SN}$, then $\mathsf{INL}_{\rho}r \in \mathsf{SN}$ and $\mathsf{INR}_{\rho}r \in \mathsf{SN}$.

(β_{+}) If $(\mathsf{INL}_{\rho}r)st\vec{s}$ is a term and $sr\vec{s} \in \mathsf{SN}$ and $\boxed{t \in \mathsf{SN}}$, then $(\mathsf{INL}_{\rho}r)st\vec{s} \in \mathsf{SN}$.
If $(\mathsf{INR}_{\rho}r)st\vec{s}$ is a term and $tr\vec{s} \in \mathsf{SN}$ and $\boxed{s \in \mathsf{SN}}$, then $(\mathsf{INR}_{\rho}r)st\vec{s} \in \mathsf{SN}$.

Every statement made in section 2.1.4 for system F (on nf, NF, wn, WN, Ω, SN and sn) is also true for eF.
We may formulate a more conspicuous version of Lemma 2.37:

Lemma 2.46 There is no closed term of type 0 in WN.

Proof Assume a closed term $r^{0} \in \mathsf{WN}$. Then $\Omega(r) \in \mathsf{NF}$, and $\Omega(r)$ is also a closed term of type 0 (due to the statement of Lemma 2.30). It is immediate from the definition of NF that this is impossible. □

If we can show that every term is in WN, we may infer that the logical system represented by eF (via Curry-Howard) is consistent (there is no closed proof of 0 which represents $\perp$). Therefore we have to expect a logically very demanding proof of $\mathsf{WN} = \Lambda$ which uses means beyond second-order propositional logic. Because the number-theoretic functions which are provably total in second-order arithmetic are representable in system eF, the proof has even to go beyond the means of second-order arithmetic (see [GLT89], chapter 15, for this result).

2.2.5 Embeddings

An embedding of a typed term rewrite system $\mathcal{S}$ into a typed term rewrite system $\mathcal{S}'$ is a function $-'$ (the $-$ sign represents the indefinite argument of the function $'$) which assigns to every type ρ of $\mathcal{S}$ a type ρ' of $\mathcal{S}'$ and to every term $r : \rho$ of $\mathcal{S}$ a term $r' : \rho'$ of $\mathcal{S}'$ such that the following implication holds: If $r \to s$ in $\mathcal{S}$, then $r' \to^{+} s'$ in $\mathcal{S}'$.
The main motivation for studying such embeddings is the inference of strong normalization for the source system $\mathcal{S}$ from strong normalization of the target system $\mathcal{S}'$. This will now be made precise.
Recall that the transitive closure $\to^{+}$ of the relation $\to$ is inductively defined by[10]:

($\to$) If $r \to s$, then $r \to^{+} s$.

(trans) If $r \to^{+} s$ and $s \to^{+} t$, then $r \to^{+} t$.

The reflexive transitive closure $\to^{*}$ of the relation $\to$ is inductively defined by:

($\to$) If $r \to s$, then $r \to^{*} s$.

(refl) $r \to^{*} r$.

(trans) If $r \to^{*} s$ and $s \to^{*} t$, then $r \to^{*} t$.

Lemma 2.47 1. $\to^{+} \subseteq \to^{*}$.

[10] In case of familiarity with this presentation one should only read the remark at the end of this section.

2. $\to^* = \to^+ \cup \{(r,r) | r \in \Lambda\}$.

3. $r \to^* s \to^+ t$ implies $r \to^+ t$.

4. $r \to^+ t$ implies that there is an s such that $r \to s \to^* t$.

Proof The first statement is trivial as the set of clauses for $\to^+$ is a subset of the set of clauses for $\to^*$. 2.: $\supseteq$: by 1. and (refl). $\subseteq$: Induction on $r \to^* s$. Cases $(\to)$ and (refl): Trivial. Case (trans): Assume $r \to^* s \to^* t$. By induction hypothesis $r \to^+ s$ or $r = s$. Also by induction hypothesis $s \to^+ t$ or $s = t$. In each of the four cases we get $r \to^+ t$ or $r = t$. 3.: Follows from 2. 4.: Induction on $r \to^+ t$. Case $(\to)$: Set $s := t$. Case (trans): Assume $r \to^+ t_0 \to^+ t$. By induction hypothesis there is an s such that $r \to s \to^* t_0$. $t_0 \to^+ t$ implies $t_0 \to^* t$ by 1. $s \to^* t_0 \to^* t$ implies $s \to^* t$ by transitivity. (One could also use 3. and 1.) □

Let sn be the set of strongly normalizing terms with respect to $\to$ as usual. Let sn^+ be the set of strongly normalizing terms with respect to $\to^+$.

Lemma 2.48 $\mathsf{sn}^+ = \mathsf{sn}$.

Proof $\subseteq$: Induction on $r \in \mathsf{sn}^+$: Assume that for all s with $r \to^+ s$ we have $s \in \mathsf{sn}$. Show $r \in \mathsf{sn}$. Let $r \to s$. It suffices to show $s \in \mathsf{sn}$. This follows because $\to \subseteq \to^+$. $\supseteq$: Induction on $r \in \mathsf{sn}$: Assume that for all s with $r \to s$ we have $s \in \mathsf{sn}^+$. Show $r \in \mathsf{sn}^+$. Let $r \to^+ t$. It suffices to show $t \in \mathsf{sn}^+$. Due to Lemma 2.47 (4) there is an s such that $r \to s \to^* t$. Hence, $s \in \mathsf{sn}^+$. Due to Lemma 2.47 (2), $s = t$ or $s \to^+ t$. The first case is trivial, in the second $t \in \mathsf{sn}^+$ because of the definition of sn^+. □

Lemma 2.49 (Main Lemma on Embeddings) Assume an embedding $-'$ of $\mathcal{S}$ in $\mathcal{S}'$. Let sn be the set of strongly normalizing terms in $\mathcal{S}$ and sn' be the set of strongly normalizing terms in $\mathcal{S}'$. Then for every term r of $\mathcal{S}$ the following holds: If $r' \in \mathsf{sn}'$, then $r \in \mathsf{sn}$.

Proof Induction on $r' \in (\mathsf{sn}')^+$, where $(\mathsf{sn}')^+$ is the set of terms in $\mathcal{S}'$ which are strongly normalizing with respect to $\to^+$: Lemma 2.48 shows that this set is the same as sn'. However, we use the induction principle of $(\mathsf{sn}')^+$: Assume that for every term t of $\mathcal{S}'$ with $r' \to^+ t$ and every term s of $\mathcal{S}$ with $s' = t$ we have $s \in \mathsf{sn}$. We have to show $r \in \mathsf{sn}$. Let $r \to s$. It suffices to show $s \in \mathsf{sn}$. The definition of embeddings implies $r' \to^+ s'$. □

Another application of Lemma 2.48 is the following which will be used in section 4.7.1:

Lemma 2.50 If $r \to^+ r$, then $r \notin \mathsf{sn}$.

Proof Show that $\Lambda \setminus \{r\}$ satisfies the defining clause of sn^+: Assume a term s such that for all s' with $s \to^+ s'$ we have $s' \neq r$. We have to show $s \neq r$. This is trivial. Hence, $\mathsf{sn} = \mathsf{sn}^+ \subseteq \Lambda \setminus \{r\}$ and consequently $r \notin \mathsf{sn}$. □

It should be noted that this lemma is trivial if sn is defined via the notion of infinite reduction sequences. The other statements of this section would have been obvious, too. Lemma 2.48 states that there is no difference between sn and sn^+. Nevertheless the statement "induction on $r \in \mathsf{sn}^+$" shall always express that induction along the inductive definition of sn^+ is carried out, i.e. "course-of-value induction".
For convenience define the binary relations $\to^n$, $n \in \mathbb{N}$, by recursion on n as follows:

$$r \to^0 s :\Leftrightarrow r = s \quad \text{and} \quad r \to^{n+1} s :\Leftrightarrow \exists t. r \to t \wedge t \to^n s.$$

2.2.6 Embedding system eF in system F

The result is essentially not new. See e.g. [GLT89, pp. 84f] for an exposition. Perhaps it is interesting to see how many details have to be checked in this section. My interest concentrates on the fact that the encoding of the types and type constructs is reduction-preserving as will be the case for most of the encodings described in this work. And because β-reduction for $\to$ and $\forall$ is term and type substitution it is natural that substitution issues and therefore also considerations on variables dominate the "details".
For every type ρ of system eF define the type ρ' of system F by recursion on ρ and simultaneously prove that $\mathsf{FV}(\rho') = \mathsf{FV}(\rho)$ as follows:

(V) $\alpha' := \alpha$.

$(\to)$ $(\rho \to \sigma)' := \rho' \to \sigma'$.

$(\forall)$ $(\forall \alpha \rho)' := \forall \alpha \rho'$.

(0) $0' := \forall \alpha \alpha$.

(1) $1' := \forall \alpha. \alpha \to \alpha$.

$(\times)$ $(\rho \times \sigma)' := \forall \alpha. (\rho' \to \sigma' \to \alpha) \to \alpha$ for $\alpha \notin \mathsf{FV}(\rho) \cup \mathsf{FV}(\sigma)$.

$(+)$ $(\rho + \sigma)' := \forall \alpha. (\rho' \to \alpha) \to (\sigma' \to \alpha) \to \alpha$ for $\alpha \notin \mathsf{FV}(\rho) \cup \mathsf{FV}(\sigma)$.

(The proofs were obvious.) This definition is compatible with the variable conventions for the systems F and eF. The rules (V), $(\to)$ and $(\forall)$ are called the homomorphic type rules for F.

Lemma 2.51 $(\rho[\vec{\alpha} := \vec{\sigma}])' = \rho'[\vec{\alpha} := \vec{\sigma}']$.

Proof Induction on ρ. □

From now on we usually omit the type of a typed term variable except at binding occurrences (that is directly behind the λ) if it is the type of the innermost binding occurrence of this untyped variable, e.g. we write $\lambda x^\alpha x$ instead of $\lambda x^\alpha x^\alpha$, but do not change $\lambda x^\alpha \lambda x^{\alpha \to \alpha} x^\alpha$ because $\lambda x^\alpha \lambda x^{\alpha \to \alpha} x$ would mean $\lambda x^\alpha \lambda x^{\alpha \to \alpha} x^{\alpha \to \alpha}$. Of course, writing such a term is no good style.
For every term r^ρ of system eF define the term r' of system F by recursion on r and simultaneously prove that $r' : \rho'$ and $\mathsf{FTV}(r') = \mathsf{FTV}(r)$ and $\mathsf{FV}(r') = \{x^{\sigma'} | x^\sigma \in \mathsf{FV}(r)\}$ as follows:

(V) $(x^\rho)' := x^{\rho'}$. Proof trivial.

($\to$-I) $(\lambda x^\rho r)' := \lambda x^{\rho'} r'$, where we may assume that for every $x^\sigma \in \mathsf{FV}(r)$ we have $\sigma = \rho$. Proof obvious.

($\to$-E) $(rs)' := r's'$. Proof obvious.

($\forall$-I) $(\Lambda \alpha r)' := \Lambda \alpha r'$. Proof obvious (Well-definedness follows from the claim on $\mathsf{FV}(r)$).

($\forall$-E) $(r\sigma)' := r'\sigma'$. Proof obvious.

(0-E) $(r\rho)' := r' \mathrm{E}_\forall \rho'$. Proof obvious.

(1-I) $\mathsf{IN1}' := \Lambda \alpha \lambda x^\alpha x$. Proof obvious.

($\times$-I) $\langle r^\rho, s^\sigma \rangle' := \Lambda \alpha \lambda z^{\rho' \to \sigma' \to \alpha}. z r' s'$ for $\alpha \notin \mathsf{FTV}(\langle r, s \rangle)$ and $z \notin \mathsf{FV}(r) \cup \mathsf{FV}(s)$. Proof obvious (use Lemma 2.8).

(×-E) $(r^{\rho\times\sigma}\mathsf{L})' := r'\mathrm{E}_\forall\rho'(\lambda x^{\rho'}\lambda y^{\sigma'}x)$ (for $x\neq y$). $(r^{\rho\times\sigma}\mathsf{R})' := r'\mathrm{E}_\forall\sigma'(\lambda x^{\rho'}\lambda y^{\sigma'}y)$ (again for $x\neq y$). Proof obvious (use Lemma 2.8).

(+-I) $(\mathsf{INL}_\sigma r^\rho)' := \Lambda\alpha\lambda x^{\rho'\to\alpha}\lambda y^{\sigma'\to\alpha}.xr'$ for $\alpha\notin\mathsf{FTV}(\mathsf{INL}_\sigma r^\rho)$ and $x,y\notin\mathsf{FV}(r)$ (with $x\neq y$). $(\mathsf{INR}_\rho r^\sigma)' := \Lambda\alpha\lambda x^{\rho'\to\alpha}\lambda y^{\sigma'\to\alpha}.yr'$ for $\alpha\notin\mathsf{FTV}(\mathsf{INR}_\rho r^\sigma)$ and also $x,y\notin\mathsf{FV}(r)$ (with $x\neq y$). Proof obvious (use Lemma 2.8).

(+-E) $(r\mathrm{E}_+\tau st)' := r'\mathrm{E}_\forall\tau's't'$. Proof obvious (use Lemma 2.8).

This definition is again compatible with the variable conventions for F and eF. The rules (V), ($\to$-I), ($\to$-E), ($\forall$-I) and ($\forall$-E) are called the homomorphic term rules for F.
Note that the condition in ($\to$-I) prevents us from setting e. g. $(\lambda x^{\forall\alpha\alpha}x^0)' := \lambda x^{\forall\alpha\alpha}x^{\forall\alpha\alpha}$.

Lemma 2.52 $(r[\vec{x}^{\vec{\rho}} := \vec{s}^{\vec{\rho}}])' = r'[\vec{x}^{\vec{\rho}'} := \vec{s}'] $ and $(r[\vec{\alpha} := \vec{\sigma}])' = r'[\vec{\alpha} := \vec{\sigma}']$.

Proof Induction on r. □

Lemma 2.53 If $r\to\hat{r}$, then $r'\to^+\hat{r}'$, i. e., $-'$ is an embedding of eF in F.

Proof Induction on $r\to\hat{r}$: The cases $(\beta_\to)$ and $(\beta_\forall)$ follow immediately from Lemma 2.52.

$$
\begin{array}{lrcl}
(\beta_\times): & (\langle r^\rho, s^\sigma\rangle\mathsf{L})' & = & (\Lambda\alpha\lambda z^{\rho'\to\sigma'\to\alpha}.zr's')\rho'(\lambda x^{\rho'}\lambda y^{\sigma'}x)\\
& & \to & (\lambda z^{\rho'\to\sigma'\to\rho'}.zr's')(\lambda x^{\rho'}\lambda y^{\sigma'}x)\\
& & \to & (\lambda x^{\rho'}\lambda y^{\sigma'}x)r's'\\
& & \to & (\lambda y^{\sigma'}r')s'\\
& & \to & r'
\end{array}
$$

We used Lemma 2.8, Corollary 2.2 and Lemma 2.16 for the first reduction step and Lemma 2.11 and $\mathsf{FV}(r') = \{x^{\sigma'}|x^\sigma\in\mathsf{FV}(r)\}$ and $\mathsf{FV}(s') = \{x^{\sigma'}|x^\sigma\in\mathsf{FV}(s)\}$ (implying $z\notin\mathsf{FV}(r')\cup\mathsf{FV}(s')$) for the second. The case with R in place of L is treated similarly.

$$
\begin{array}{lrcl}
(\beta_+): & ((\mathsf{INL}_\sigma r^\rho)s^{\rho\to\tau}t^{\sigma\to\tau})' & = & (\Lambda\alpha\lambda x^{\rho'\to\alpha}\lambda y^{\sigma'\to\alpha}.xr')\tau's't'\\
& & \to & (\lambda x^{\rho'\to\tau'}\lambda y^{\sigma'\to\tau'}.xr')s't'\\
& & \to & (\lambda y^{\sigma'\to\tau'}.s'r')t' \quad \text{where we assume that } y^{\sigma'\to\tau'}\notin\mathsf{FV}(s')\\
& & \to & s'r' = (sr)'
\end{array}
$$

We used the same lemmas as for the case $(\beta_\times)$. The case with INR in place of INL is treated similarly.
The other clauses are easily proved by using the induction hypothesis. □

2.3 Extension by infimum types

Infimum types are introduced to give a clear understanding of the embedding of monotone inductive types with iteration (and no recursion, more specifically: of $\mathsf{varEMIT{-}rec}$) into eF carried out in section 4.3.3.
The following motivation[11] could be made clear by using modified realizability as indicated in the introduction.
Let $(U,\leq)$ be a complete lattice, i. e., U is a non-empty set and $\leq$ is a partial ordering on U ($\leq$ is a reflexive, antisymmetric and transitive relation on U) and for every $\mathcal{M}\subseteq U$ there is an infimum (greatest lower bound, meet) $\bigwedge\mathcal{M}$ and a supremum (least upper bound, join) $\bigvee\mathcal{M}$ in U. It is well-known that for $\mathcal{M}\subseteq U$, we have

$$\bigvee\mathcal{M} = \bigwedge\{M\in U|\forall M'\in\mathcal{M}.M'\leq M\},$$

[11] I got my motivation for this extension from reading in [Tat94] on greatest lower bound inductive definitions which in my setting would be nothing else than studying $i\alpha.\rho\to\alpha$ instead of $\mu\alpha\rho$. I preferred to study general $i\alpha\rho$ and later (in section 4.3.3) embed $\mathsf{varEMIT{-}rec}$ into IT by only using infimum types of the above form.

i. e., the supremum of a set is[12] the infimum of all upper bounds of that set. Hence, we may exclusively concentrate on infima.
There are two defining properties of infima: $I = \bigwedge\mathcal{M}$ iff $\forall M\in\mathcal{M}.I\leq M$ (I is a lower bound of $\mathcal{M}$) and $\forall M\in U.(\forall M'\in\mathcal{M}.M\leq M')\Rightarrow M\leq I$ (I is maximal with respect to being lower bound of $\mathcal{M}$).
Hence, we have two rules for $\bigwedge\mathcal{M}$:

($\bigwedge$-I) $\forall M\in U.(\forall M'\in\mathcal{M}.M\leq M')\Rightarrow M\leq\bigwedge\mathcal{M}$

($\bigwedge$-E) $\forall M\in\mathcal{M}.\bigwedge\mathcal{M}\leq M$

Now let $(U,\subseteq)$ be a complete lattice of sets with set inclusion as partial ordering ($\subseteq$ will always denote set inclusion also in the later chapters) and let $|U| := \bigwedge\emptyset$ be its underlying set of elements. Let x, y always denote elements of $|U|$ and M, M' elements of U.
The (implicitly universally quantified) rules now read (after some rearrangement)

($\bigwedge$-I) $(\forall M'.M'\in\mathcal{M}\Rightarrow\forall y(y\in M\Rightarrow y\in M'))\Rightarrow x\in M\Rightarrow x\in\bigwedge\mathcal{M}$

($\bigwedge$-E) $x\in\bigwedge\mathcal{M}\Rightarrow M\in\mathcal{M}\Rightarrow x\in M$

Note that we associate $\Rightarrow$ to the right as we do for $\to$ between types.
This is the motivation for the infimum types to be defined in this section. We model elements of U and also statements on elements of U by types. Model "$M\in\mathcal{M}$" by $\rho[\alpha:=\tau]$, if $\mathcal{M}$ is modelled by $\lambda\alpha\rho$ and M by τ. We introduce a new type $i\alpha\rho$ to model $\bigwedge\mathcal{M}$. It will always be the case that elements of $|U|$ are irrelevant to the interpretation of lattice-theoretic concepts in our typed systems. We try to interpret the rule ($\bigwedge$-I): Model M by τ. The quantifier over elements of U is interpreted by $\forall$ and hence the truth of $(\forall M'.M'\in\mathcal{M}\Rightarrow\forall y(y\in M\Rightarrow y\in M'))$ by some term of type $\forall\alpha.\rho\to\tau\to\alpha$ (of course with $\alpha\notin\mathsf{FV}(\tau)$) and the truth of $x\in M$ by a term of type τ. We infer the truth of $x\in\bigwedge\mathcal{M}$ which will be modelled by some term of type $i\alpha\rho$ constructed out of the two terms. Hence, we have the rule of infimum introduction

(i-I) If $\ell^{\forall\alpha.\rho\to\tau\to\alpha}\in\Lambda$ and $t^\tau\in\Lambda$ (with $\alpha\notin\mathsf{FV}(\tau)$) then $(C_{i\alpha\rho}\ell t)^{i\alpha\rho}\in\Lambda$.

Now interpret ($\bigwedge$-E): Model M by σ. The truth of $x\in\bigwedge\mathcal{M}$ is interpreted by a term of type $i\alpha\rho$ and the truth of $M\in\mathcal{M}$ by a term of type $\rho[\alpha:=\sigma]$. We infer the truth of $x\in M$, which will be modelled by a construction of a term of type σ out of the two other terms. This gives the rule of infimum elimination

(i-E) If $r^{i\alpha\rho}\in\Lambda$ and $s^{\rho[\alpha:=\sigma]}\in\Lambda$, then $(rs)^\sigma\in\Lambda$.

2.3.1 Extension of the type system by infimum types

The type system of the extension by infimum types is defined by adding the following clause to the inductive definition of Types for eF:

(i) If $\alpha\in\mathsf{Typevars}$ and $\rho\in\mathsf{Types}$, then $i\alpha\rho\in\mathsf{Types}$.

The variable α in $i\alpha\rho$ is considered to be bound by i in exactly the same way as α is bound in $\forall\alpha\rho$ by $\forall$. Hence, the renaming convention for bound type variables in types will be extended accordingly. i is assumed to syntactically bind stronger than $\to$, $\times$ and $+$, as was done for $\forall$.
The definition of $\mathsf{FV}(\rho)$ for system eF is extended by the following clause:

[12] This fact may be spelled out constructively: The system of supremum types motivated in the same fashion may be embedded into the system of infimum types and vice versa. Note that this includes preservation of β-reduction for infimum and supremum types.

(i) $\mathsf{FV}(i\alpha\rho) := \mathsf{FV}(\rho) \setminus \{\alpha\}$.

The definition remains compatible with the renaming convention.
The definition of $\rho[\vec{\alpha} := \vec{\sigma}]$ is extended by the following clause:

(i) $(i\alpha\rho)[\vec{\alpha} := \vec{\sigma}] := i\alpha.\rho[\vec{\alpha} := \vec{\sigma}]$, where we assume that $\alpha \notin \vec{\alpha} \cup \mathsf{FV}(\vec{\sigma})$.

It is again compatible with the renaming convention.
We adopt the same naming convention.
The statements on types given for system F are also true in this extension as for the time being, i and $\forall$ are only different because of different names. If one added e. g. the existential quantifier $\exists$ to the system (see section 2.4.2) the same procedure would have to take place. This suggests introducing the concept of abstracting a type variable α in a type ρ the result of which is denoted by $\lambda\alpha\rho$. The λ hints at our variable convention because it is traditional to adopt this convention whenever one deals with λ-abstraction. $\forall$, i and $\exists$ would then simply be constants which applied to an abstracted type give another type, e. g. $\forall\alpha\rho$ would be replaced by $\forall(\lambda\alpha\rho)$. I decided not to use such a framework, but nevertheless have to introduce the concept of abstracted types for dealing with terms of positive inductive types (which are defined in section 3.2).

2.3.2 The term rewrite system IT of infimum types

The (second) definition of typed terms and their free variables for system eF is extended by the following clauses:

(i-I) If $\ell^{\forall\alpha.\rho\to\tau\to\alpha} \in \Lambda$ and $t^\tau \in \Lambda$ (with $\alpha \notin \mathsf{FV}(\tau)$)[13], then $(C_{i\alpha\rho}\ell t)^{i\alpha\rho} \in \Lambda$ and $\mathsf{FV}(C_{i\alpha\rho}\ell t) := \mathsf{FV}(\ell) \cup \mathsf{FV}(t)$.

(i-E) If $r^{i\alpha\rho} \in \Lambda$ and $s^{\rho[\alpha:=\sigma]} \in \Lambda$, then $(rs)^\sigma \in \Lambda$ (also written as $r\mathrm{E}_i\sigma s$ or shorter $r\mathrm{E}_i s$) and $\mathsf{FV}(rs) := \mathsf{FV}(r) \cup \mathsf{FV}(s)$.

This definition is again compatible with the renaming convention for bound type variables.
If $r : i\alpha\rho$, we say that r is of infimum type. We extend the naming convention by assuming that ℓ always denotes a term.
If $\alpha \in \mathsf{FV}(\rho)$ then σ can be read off $\rho[\alpha := \sigma]$ and therefore the type of the term rs is determined by r and s. Otherwise rs can have any type. Because we want to have the uniqueness of the type of a term (Lemma 2.7) we have to take the notation $r\mathrm{E}_i\sigma s$ as the "true" term. However, in theoretic studies on the system IT it is quite unusual to focus specifically on cases where $\alpha \notin \mathsf{FV}(\rho)$ and therefore the appearance of σ in the string which represents the type of $s^{\rho[\alpha:=\sigma]}$ clearly indicates the type σ of rs.
Extend the recursive definition of $\mathsf{FTV}(r)$ by the following clauses:

(i-I) $\mathsf{FTV}(C_{i\alpha\rho}\ell t) := \mathsf{FV}(i\alpha\rho) \cup \mathsf{FTV}(\ell) \cup \mathsf{FTV}(t)$.

(i-E) $\mathsf{FTV}(rs) := \mathsf{FTV}(r) \cup \mathsf{FV}(\sigma) \cup \mathsf{FTV}(s)$ (Recall the convention on the type σ of rs).

[13] Note that without this condition the renaming convention for bound type variables would fail. Therefore, conditions of this kind may be left implicit by tacitly assuming that every definition has an appropriate reading which is compatible with the variable convention and by taking this reading for the "true" definition. Nevertheless, the important side conditions will always be indicated in parentheses.
Because of $\alpha \notin \mathsf{FV}(\tau)$ one could also require that $\ell : \tau \to \forall\alpha.\rho \to \alpha$. But then there would be no point in using this ℓ and t^τ and we could use the rule $\ell : \forall\alpha.\rho \to \alpha \Rightarrow C_{i\alpha\rho}\ell : i\alpha\rho$ instead which would result in the embedding into F becoming even more trivial. The reason for keeping to the original definition is the exact correspondence with the lattice-theoretic reasoning.

The definition of $r[\vec{x}^{\vec{\rho}} := \vec{s}]$ together with the proof of $r^\rho[\vec{x}^{\vec{\rho}} := \vec{s}] : \rho$ and $\mathsf{FV}(r[\vec{x}^{\vec{\rho}} := \vec{s}]) = (\mathsf{FV}(r) \setminus \vec{x}^{\vec{\rho}}) \cup \bigcup\{\mathsf{FV}(s_i) | x_i^{\rho_i} \in \mathsf{FV}(r) \wedge x_i^{\rho_i} \neq x_j^{\rho_j} \text{ for } j < i\}$ (i. e., the statement of Lemma 2.10) is extended by the following clauses:

(i-I) $(C_{i\alpha\rho}\ell t)[\vec{x}^{\vec{\rho}} := \vec{s}] := C_{i\alpha\rho}\ell[\vec{x}^{\vec{\rho}} := \vec{s}]t[\vec{x}^{\vec{\rho}} := \vec{s}]$. Proof obvious.

(i-E) $(rs)[\vec{x}^{\vec{\rho}} := \vec{s}] := r[\vec{x}^{\vec{\rho}} := \vec{s}]\mathrm{E}_i s[\vec{x}^{\vec{\rho}} := \vec{s}]$. Proof obvious.

The definition of $r[\vec{\alpha} := \vec{\sigma}]$ together with the proof of $r^\rho[\vec{\alpha} := \vec{\sigma}] : \rho[\vec{\alpha} := \vec{\sigma}]$ and $\mathsf{FV}(r[\vec{\alpha} := \vec{\sigma}]) = \{y^{\tau[\vec{\alpha}:=\vec{\sigma}]} | y^\tau \in \mathsf{FV}(r)\}$ is extended by the following clauses:

(i-I) $(C_{i\alpha\rho}\ell t)[\vec{\alpha} := \vec{\sigma}] := C_{(i\alpha\rho)[\vec{\alpha}:=\vec{\sigma}]}\ell[\vec{\alpha} := \vec{\sigma}]t[\vec{\alpha} := \vec{\sigma}]$. Proof obvious.

(i-E) $(rs)[\vec{\alpha} := \vec{\sigma}] := r[\vec{\alpha} := \vec{\sigma}]\mathrm{E}_i(\sigma[\vec{\alpha} := \vec{\sigma}])s[\vec{\alpha} := \vec{\sigma}]$. Proof obvious.

The statements made in section 2.1.2 are valid in IT, too.
The general definition of immediate subterms of a term gives the following extension for system IT: $C_{i\alpha\rho}\ell t$ has exactly the immediate subterms ℓ and t and $r\mathrm{E}_i s$ has exactly the immediate subterms r and s.
Extend the inductive definition of $r\vec{s}$ for system eF as follows:

(i-E) If $r\vec{s} : i\alpha\rho$ and $s : \rho[\alpha := \sigma]$, then $r(\vec{s}, s) := (r\vec{s})s$.

The statement of Lemma 2.23 holds also true for IT.

Lemma 2.54 (Head form) Every term has either exactly one of the forms given in Lemma 2.44 or exactly one of the following forms:

(i-I) $C_{i\alpha\rho}\ell t$

(i-R) $(C_{i\alpha\rho}\ell t)s\vec{s}$

Proof Induction on terms. □

Extend the inductive definition of the set NF for system eF by the following clause:

(i-I) If $C_{i\alpha\rho}\ell t$ is a term and $\ell \in \mathsf{NF}$ and $t \in \mathsf{NF}$, then $C_{i\alpha\rho}\ell t \in \mathsf{NF}$.

The statements made in section 2.1.4 are all valid in IT.
Define the relation $\mapsto$ as for system eF, but add the following clause:

$$(\beta_i) \qquad (C_{i\alpha\rho}\ell t)\mathrm{E}_i\sigma s \mapsto \ell\sigma st$$

As for system eF the object on the right side is automatically a term if the respective object on the left is a term (note that $\tau[\alpha := \sigma] = \tau$), and the types are equal.
The relation $\to_{whd}$ is defined as for system F and has the same properties as stated there.
Extend the inductive definition of the relation $\to$ for system eF by the following clauses (and prove that if $r \to r'$, then r and r' have the same type and $\mathsf{FV}(r') \subseteq \mathsf{FV}(r)$):

(i-I) If $\ell \to \ell'$, then $C_{i\alpha\rho}\ell t \to C_{i\alpha\rho}\ell' t$. If $t \to t'$, then $C_{i\alpha\rho}\ell t \to C_{i\alpha\rho}\ell t'$.

(i-E) If $r \to r'$, then $r\mathrm{E}_i s \to r'\mathrm{E}_i s$. If $s \to s'$, then $r\mathrm{E}_i s \to r\mathrm{E}_i s'$.

(The proofs were trivial.)
Extend the inductive definition of WN for system eF by the following clauses:

(i-I) If $C_{i\alpha\rho}\ell t$ is a term and $\ell \in \mathsf{WN}$ and $t \in \mathsf{WN}$, then $C_{i\alpha\rho}\ell t \in \mathsf{WN}$.

(β_i) If $(C_{i\alpha\rho}\ell t)\mathrm{E}_i\sigma s\vec{s}$ is a term and $\ell\sigma st\vec{s} \in \mathsf{WN}$, then $(C_{i\alpha\rho}\ell t)\mathrm{E}_i\sigma s\vec{s} \in \mathsf{WN}$.

Again definition by recursion on WN is admissible. Extend the recursive definition of the function Ω from WN to Λ and the simultaneous proof of $r \to^* \Omega(r) \in \mathsf{NF}$ for $r \in \mathsf{WN}$ by the following clauses:

(i-I) $\Omega(C_{i\alpha\rho}\ell t) := C_{i\alpha\rho}\Omega(\ell)\Omega(t)$.

(β_i) $\Omega((C_{i\alpha\rho}\ell t)\mathrm{E}_i\sigma s\vec{s}) := \Omega(\ell\sigma st\vec{s})$.

The proofs are always obvious.
Extend the definition of SN for system eF by the following clauses:

(i-I) If $C_{i\alpha\rho}\ell t$ is a term and $\ell \in \mathsf{SN}$ and $t \in \mathsf{SN}$, then $C_{i\alpha\rho}\ell t \in \mathsf{SN}$.

(β_i) If $(C_{i\alpha\rho}\ell t)\mathrm{E}_i\sigma s\vec{s}$ is a term and $\ell\sigma st\vec{s} \in \mathsf{SN}$, then $(C_{i\alpha\rho}\ell t)\mathrm{E}_i\sigma s\vec{s} \in \mathsf{SN}$.

The only difference between the i-clauses for SN and WN is the name of the defined set.
Every statement made in section 2.1.4 for system F (on nf, NF, wn, WN, Ω, SN and sn) is also true for IT.

2.3.3 The embedding of system IT in system eF

As a motivation let $(U, \subseteq)$ be a complete lattice where the infimum is always given by set intersection, i. e., for $\mathcal{M} \subseteq U$, $\bigwedge\mathcal{M} = \bigcap\mathcal{M} = \{x | \forall M \in U.M \in \mathcal{M} \Rightarrow x \in M\}$. (We use the same conventions as in the introduction to infimum types.) Hence

$$x \in \bigwedge\mathcal{M} \Leftrightarrow \forall M.(M \in \mathcal{M} \Rightarrow x \in M)$$

We immediately read off this equivalence the interpretation of $i\alpha\rho$ by $\forall\alpha.\rho \to \alpha$. Now we have to check that the definition based on this idea is also compatible with β-reduction for all type constructs including β_i.
For every type ρ of system IT define the type ρ' of system eF by recursion on ρ and simultaneously prove that $\mathsf{FV}(\rho') = \mathsf{FV}(\rho)$ as follows:

(F) The homomorphic type rules for F.

($\circ$) $(\rho \circ \sigma)' := \rho' \circ \sigma'$ for $\circ \in \{\times, +\}$.

(0) $0' := 0$.

(1) $1' := 1$.

(i) $(i\alpha\rho)' := \forall\alpha.\rho' \to \alpha$.

(The proofs were obvious.) This definition is compatible with the variable conventions for the systems IT and eF. The rules (F), ($\circ$), (0) and (1) are called the homomorphic type rules for eF.

Lemma 2.55 $(\rho[\vec{\alpha} := \vec{\sigma}])' = \rho'[\vec{\alpha} := \vec{\sigma}']$.

Proof Induction on ρ. □

For every term r^ρ of system IT define the term r' of system eF by recursion on r and simultaneously prove that $r' : \rho'$ and $\mathsf{FTV}(r') = \mathsf{FTV}(r)$ and $\mathsf{FV}(r') = \{x^{\sigma'} | x^\sigma \in \mathsf{FV}(r)\}$ as follows:

(F) The homomorphic term rules for F.

(0-E) $(r\rho)' := r'\rho'$.

(1-I) $\mathsf{IN1}' := \mathsf{IN1}$.

($\times$-I) $\langle r, s\rangle' := \langle r', s'\rangle$.

($\times$-E) $(r\mathsf{L})' := r'\mathsf{L}$. $(r\mathsf{R})' := r'\mathsf{R}$.

($+$-I) $(\mathsf{INL}_\rho r)' := \mathsf{INL}_{\rho'} r'$. $(\mathsf{INR}_\rho r)' := \mathsf{INR}_{\rho'} r'$.

($+$-E) $(r\mathrm{E}_+\tau st)' := r'\mathrm{E}_+\tau's't'$.

(i-I) $(C_{i\alpha\rho}\ell^{\forall\alpha.\rho\to\tau\to\alpha}t^\tau)' := \Lambda\alpha\lambda x^{\rho'}.\ell'\alpha xt'$, where we assume that $\alpha \notin \mathsf{FTV}(\ell) \cup \mathsf{FTV}(t)$ and require $x \notin \mathsf{FV}(\ell) \cup \mathsf{FV}(t)$.

(i-E) $(r^{i\alpha\rho}\mathrm{E}_i s^{\rho[\alpha:=\sigma]})' := r'\mathrm{E}_\forall\sigma's'$.

(The proofs were always obvious. (i-E) needs the previous lemma for type correctness.) This definition is again compatible with the variable conventions for IT and eF. The rules (F), (0-E), (1-I), ($\times$-I), ($\times$-E), ($+$-I) and ($+$-E) are called the homomorphic term rules for eF.

Lemma 2.56 $(r[\vec{x}^{\vec{\rho}} := \vec{s}^{\vec{\rho}}])' = r'[\vec{x}^{\vec{\rho}'} := \vec{s}']$ and $(r[\vec{\alpha} := \vec{\sigma}])' = r'[\vec{\alpha} := \vec{\sigma}']$.

Proof Induction on r. □

Lemma 2.57 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 Lemma 2.56, ($\beta_\times$) and (β_+) are trivial.
(β_i): $$\begin{array}{rcl}((C_{i\alpha\rho}\ell^{\forall\alpha.\rho\to\tau\to\alpha}t^\tau)\mathrm{E}_i s^{\rho[\alpha:=\sigma]})' & = & (\Lambda\alpha\lambda x^{\rho'}.\ell'\alpha xt')\mathrm{E}_\forall\sigma's' \\ & \to & (\lambda x^{\rho'[\alpha:=\sigma']}.\ell'\sigma'xt')s' \\ & \to & \ell'\sigma's't' = (\ell\sigma st)'\end{array}$$
We used Lemma 2.16 for the first reduction step and Lemma 2.11 for the second (and $x \notin \mathsf{FV}(\ell) \cup \mathsf{FV}(t) \Rightarrow x \notin \mathsf{FV}(\ell') \cup \mathsf{FV}(t')$).
The other clauses are easily proved by using the induction hypothesis. □

2.4 Extension by the existential quantifier

Types of the form $\exists\alpha\rho$ are added to eF. It is well-known that $\exists\alpha\rho$ may be coded via universal quantification. It is even reduction-preserving. As was the case for the extension of F to eF, this does not carry over to every extension by μ-types, does not respect η-reduction and does not allow for permutative conversions (for $\exists$). Strict positivity is also not preserved—already a justification for this extension.

2.4.1 Extension of the type system by existential types

The type system of the extension by existential types is defined by adding the following clause to the inductive definition of Types for eF:

($\exists$) If $\alpha \in \mathsf{Typevars}$ and $\rho \in \mathsf{Types}$, then $\exists\alpha\rho \in \mathsf{Types}$.

As was the case for i and $\forall$, the variable α in $\exists\alpha\rho$ is considered to be bound by $\exists$. Hence, the renaming convention for bound type variables in types will be extended accordingly. $\exists$ is assumed to syntactically bind stronger than $\to$, $\times$ and $+$, as was done for $\forall$ (and also for i). The definition of $\mathsf{FV}(\rho)$ for system eF is extended by the following clause:

($\exists$) $\mathsf{FV}(\exists\alpha\rho) := \mathsf{FV}(\rho) \setminus \{\alpha\}$.

The definition of $\rho[\vec{\alpha} := \vec{\sigma}]$ is extended by the following clause:

($\exists$) $(\exists\alpha\rho)[\vec{\alpha} := \vec{\sigma}] := \exists\alpha.\rho[\vec{\alpha} := \vec{\sigma}]$, where we assume that $\alpha \notin \vec{\alpha} \cup \mathsf{FV}(\vec{\sigma})$.

Again $\exists$ and $\forall$ are only different because of their different names and therefore the statements in section 2.1.1 remain true in this extension.

2.4.2 The extension eF+ex of system eF by existential types

The (second) definition of typed terms and their free variables for system eF is extended by the following clauses:

($\exists$-I) If $t^{\rho[\alpha:=\tau]} \in \Lambda$, then $(C_{\exists\alpha\rho,\tau}t)^{\exists\alpha\rho} \in \Lambda$ and $\mathsf{FV}(C_{\exists\alpha\rho,\tau}t) := \mathsf{FV}(t)$.

($\exists$-E) If $r^{\exists\alpha\rho} \in \Lambda$ and $s^{\forall\alpha.\rho\to\sigma} \in \Lambda$ (with $\alpha \notin \mathsf{FV}(\sigma)$), then $(rs)^\sigma \in \Lambda$ (also written as $r\mathrm{E}_\exists\sigma s$ or shorter $r\mathrm{E}_\exists s$) and $\mathsf{FV}(rs) := \mathsf{FV}(r) \cup \mathsf{FV}(s)$.

This definition is again compatible with the renaming convention for bound type variables.
If $r : \exists\alpha\rho$, we say that r is of existential type.
Extend the recursive definition of $\mathsf{FTV}(r)$ by the following clauses:

($\exists$-I) $\mathsf{FTV}(C_{\exists\alpha\rho,\tau}t) := \mathsf{FV}(\exists\alpha\rho) \cup \mathsf{FV}(\tau) \cup \mathsf{FTV}(t)$.

($\exists$-E) $\mathsf{FTV}(rs) := \mathsf{FTV}(r) \cup \mathsf{FTV}(s)$

The definition of $r[\vec{x}^{\vec{\rho}} := \vec{s}]$ together with the proof of $r^\rho[\vec{x}^{\vec{\rho}} := \vec{s}] : \rho$ and $\mathsf{FV}(r[\vec{x}^{\vec{\rho}} := \vec{s}]) = (\mathsf{FV}(r) \setminus \vec{x}^{\vec{\rho}}) \cup \bigcup\{\mathsf{FV}(s_i) | x_i^{\rho_i} \in \mathsf{FV}(r) \wedge x_i^{\rho_i} \neq x_j^{\rho_j} \text{ for } j < i\}$ (i.e., the statement of Lemma 2.10) is extended by the following clauses:

($\exists$-I) $(C_{\exists\alpha\rho,\tau}t)[\vec{x}^{\vec{\rho}} := \vec{s}] := C_{\exists\alpha\rho,\tau}t[\vec{x}^{\vec{\rho}} := \vec{s}]$. Proof obivous.

($\exists$-E) $(rs)[\vec{x}^{\vec{\rho}} := \vec{s}] := r[\vec{x}^{\vec{\rho}} := \vec{s}]\mathrm{E}_\exists s[\vec{x}^{\vec{\rho}} := \vec{s}]$. Proof obvious.

The definition of $r[\vec{\alpha} := \vec{\sigma}]$ together with the proof of $r^\rho[\vec{\alpha} := \vec{\sigma}] : \rho[\vec{\alpha} := \vec{\sigma}]$ and $\mathsf{FV}(r[\vec{\alpha} := \vec{\sigma}]) = \{y^{\tau[\vec{\alpha}:=\vec{\sigma}]} | y^\tau \in \mathsf{FV}(r)\}$ is extended by the following clauses:

($\exists$-I) $(C_{\exists\alpha\rho,\tau}t)[\vec{\alpha} := \vec{\sigma}] := C_{(\exists\alpha\rho)[\vec{\alpha}:=\vec{\sigma}],\tau[\vec{\alpha}:=\vec{\sigma}]}t[\vec{\alpha} := \vec{\sigma}]$. Proof obvious.

($\exists$-E) $(rs)[\vec{\alpha} := \vec{\sigma}] := r[\vec{\alpha} := \vec{\sigma}]\mathrm{E}_\exists s[\vec{\alpha} := \vec{\sigma}]$. Proof obvious.

The statements made in section 2.1.2 are valid in eF+ex, too.
The general definition of immediate subterms of a term gives the following extension for system eF+ex: $C_{\exists\alpha\rho,\tau}t$ has exactly the immediate subterm t and $r\mathrm{E}_\exists s$ has exactly the immediate subterms r and s.
Extend the inductive definition of $r\vec{s}$ for system eF as follows:

($\exists$-E) If $r\vec{s} : \exists\alpha\rho$ and $s : \forall\alpha.\rho \to \sigma$ (with $\alpha \notin \mathsf{FV}(\sigma)$), then $r(\vec{s}, s) := (r\vec{s})s$.

The statement of Lemma 2.23 holds also true for eF+ex.

Lemma 2.58 (Head form) Every term has either exactly one of the forms given in Lemma 2.44 or exactly one of the following forms:

($\exists$-I) $C_{\exists\alpha\rho,\tau}t$

($\exists$-R) $(C_{\exists\alpha\rho,\tau}t)s\vec{s}$

Proof Induction on terms. □

Extend the inductive definition of the set NF for system eF by the following clause:

($\exists$-I) If $C_{\exists\alpha\rho,\tau}t$ is a term and $t \in \mathsf{NF}$, then $C_{\exists\alpha\rho,\tau}t \in \mathsf{NF}$.

The statements made in section 2.1.4 are all valid in eF+ex.
Define the relation $\mapsto$ as for system eF, but add the following clause:

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

As for system eF the object on the right side is automatically a term if the respective object on the left is a term (note that $\sigma[\alpha := \tau] = \sigma$), and the types are equal.
The relation $\to_{whd}$ is defined as for system F and has the same properties as stated there.
Extend the inductive definition of the relation $\to$ for system eF by the following clauses (and prove that if $r \to r'$, then r and r' have the same type and $\mathsf{FV}(r') \subseteq \mathsf{FV}(r)$):

($\exists$-I) If $t \to t'$, then $C_{\exists\alpha\rho,\tau}t \to C_{\exists\alpha\rho,\tau}t'$.

($\exists$-E) If $r \to r'$, then $r\mathrm{E}_\exists s \to r'\mathrm{E}_\exists s$. If $s \to s'$, then $r\mathrm{E}_\exists s \to r\mathrm{E}_\exists s'$.

(The proofs were trivial.)
Extend the inductive definition of WN for system eF by the following clauses:

($\exists$-I) If $C_{\exists\alpha\rho,\tau}t$ is a term and $t \in \mathsf{WN}$, then $C_{\exists\alpha\rho,\tau}t \in \mathsf{WN}$.

($\beta_\exists$) If $(C_{\exists\alpha\rho,\tau}t)\mathrm{E}_\exists s\vec{s}$ is a term and $s\tau t\vec{s} \in \mathsf{WN}$, then $(C_{\exists\alpha\rho,\tau}t)\mathrm{E}_\exists s\vec{s} \in \mathsf{WN}$.

Again definition by recursion on WN is admissible. Extend the recursive definition of the function Ω from WN to Λ and the simultaneous proof of $r \to^* \Omega(r) \in \mathsf{NF}$ for $r \in \mathsf{WN}$ by the following clauses:

($\exists$-I) $\Omega(C_{\exists\alpha\rho,\tau}t) := C_{\exists\alpha\rho,\tau}\Omega(t)$.

($\beta_\exists$) $\Omega((C_{\exists\alpha\rho,\tau}t)\mathrm{E}_\exists s\vec{s}) := \Omega(s\tau t\vec{s})$.

The proofs are always obvious.
Extend the definition of SN for system eF by the following clauses:

($\exists$-I) If $C_{\exists\alpha\rho,\tau}t$ is a term and $t \in \mathsf{SN}$, then $C_{\exists\alpha\rho,\tau}t \in \mathsf{SN}$.

($\beta_\exists$) If $(C_{\exists\alpha\rho,\tau}t)\mathrm{E}_\exists s\vec{s}$ is a term and $s\tau t\vec{s} \in \mathsf{SN}$, then $(C_{\exists\alpha\rho,\tau}t)\mathrm{E}_\exists s\vec{s} \in \mathsf{SN}$.

The only difference between the $\exists$-clauses for SN and WN is the name of the defined set.
Every statement made in section 2.1.4 for system F (on nf, NF, wn, WN, Ω, SN and sn) is also true for eF+ex.

2.4.3 The embedding of system eF+ex in system eF

The essence[14] of this section may be found in [GLT89, p. 86].
For every type ρ of system $\mathsf{eF{+}ex}$ define the type ρ' of system eF by recursion on ρ and simultaneously prove that $\mathsf{FV}(\rho') = \mathsf{FV}(\rho)$ as follows:

(eF) The homomorphic type rules for eF.

($\exists$) $(\exists\alpha\rho)' := \forall\beta.(\forall\alpha.\rho' \to \beta) \to \beta$ for $\beta \notin \{\alpha\} \cup \mathsf{FV}(\rho)$.

(The proofs were obvious.) This definition is compatible with the variable conventions for the systems eF+ex and eF.

Lemma 2.59 $(\rho[\vec{\alpha} := \vec{\sigma}])' = \rho'[\vec{\alpha} := \vec{\sigma}']$.

Proof Induction on ρ. □

For every term r^ρ of system eF+ex define the term r' of system eF by recursion on r and simultaneously prove that $r' : \rho'$ and $\mathsf{FTV}(r') = \mathsf{FTV}(r)$ and $\mathsf{FV}(r') = \{x^{\sigma'} | x^\sigma \in \mathsf{FV}(r)\}$ as follows:

(eF) The homomorphic term rules for eF.

($\exists$-I) $(C_{\exists\alpha\rho,\tau} t^{\rho[\alpha:=\tau]})' := \Lambda\beta\lambda x^{\forall\alpha.\rho'\to\beta}.x\tau' t'$ for $\beta \notin \mathsf{FV}(\tau) \cup \mathsf{FTV}(t)$ and $x \notin \mathsf{FV}(t)$.

($\exists$-E) $(r^{\exists\alpha\rho}\mathrm{E}_\exists s^{\forall\alpha.\rho\to\sigma})' := r'\mathrm{E}_\forall \sigma' s'$.

(The proofs were always obvious.) This definition is again compatible with the variable conventions for eF+ex and eF.
Note that the classical interpretation of the existential quantifier via $(\exists\alpha\rho)' := (\forall\alpha.\rho' \to 0) \to 0$ would be too weak because it does not allow the interpretation of the existential elimination.

Lemma 2.60 $(r[\vec{x}^{\vec{\rho}} := \vec{s}^{\vec{\rho}}])' = r'[\vec{x}^{\vec{\rho}'} := \vec{s}'\,]$ and $(r[\vec{\alpha} := \vec{\sigma}])' = r'[\vec{\alpha} := \vec{\sigma}']$.

Proof Induction on r. □

Lemma 2.61 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 Lemma 2.60, $(\beta_\times)$ and (β_+) are trivial.
$(\beta_\exists)$:

$$\begin{aligned} ((C_{\exists\alpha\rho,\tau}t^\tau)\mathrm{E}_\exists s^{\forall\alpha.\rho\to\sigma})' &= (\Lambda\beta\lambda x^{\forall\alpha.\rho'\to\beta}.x\tau' t')\mathrm{E}_\forall\sigma' s' \\ &\to (\lambda x^{\forall\alpha.\rho'\to\sigma'}.x\tau' t')s' \\ &\to s'\tau' t' = (s\tau t)' \end{aligned}$$

We used Lemma 2.16 and Corollary 2.2 for the first reduction step and Lemma 2.11 for the second.
The other clauses are easily proved by using the induction hypothesis. □

[14] See the introductory remark to section 2.2.6.

Chapter 3

Extension of the type system by inductive types (i. e., μ-types)

Let $(U, \leq)$ be a complete lattice and $\Phi : U \to U$ be monotone[1]. By Tarski's [Tar55] fixed-point theorem Φ has a least fixed point[2] $\mathsf{lfp}(\Phi)$ which is given by

$$\mathsf{lfp}(\Phi) = \bigwedge\{M \in U | \Phi(M) \leq M\}.$$

It will be important to give and analyze the proof of this theorem in section 4.3.3.
$M \in U$ is a pre-fixed-point of Φ is defined to mean $\Phi(M) \leq M$. By definition, we have that $\mathsf{lfp}(\Phi) \leq M$ for all pre-fixed-points M of Φ, hence

(lfp-E) $\Phi(M) \leq M \Rightarrow \mathsf{lfp}(\Phi) \leq M$.

Let now $(U, \subseteq)$ be a complete lattice of sets and Φ a monotone operator on it. The last remark now reads:

(lfp-E) If $x \in \mathsf{lfp}(\Phi)$ and $\Phi(M) \subseteq M$, then $x \in M$.

This is simply the rule of induction on $\mathsf{lfp}(\Phi)$ which hence is seen as the set inductively defined by Φ (see e. g. [Acz77]).
In the systems of inductive types $\mathsf{lfp}(\Phi)$ is modelled by a type $\mu\alpha\rho$ where Φ is represented by some $\lambda\alpha\rho$, i.e. a type ρ with a selected type variable α to indicate the argument of Φ. All the systems with exception of those in the spirit of Mendler (in chapter 6) need $\lambda\alpha\rho$ to be monotone in some sense. The systems of monotone inductive types need some "monotonicity witness", i. e., a term of type $\forall\alpha\forall\beta.(\alpha \to \beta) \to \rho \to \rho[\alpha := \beta]$ and the systems of positive inductive types have the syntactic restriction of positivity on the formation rule for types of the form $\mu\alpha\rho$ which in turn ensures the existence of canonical monotonicity witnesses. The systems à la Mendler do not have any restriction on forming μ-types but their interpretation is the least fixed point of some canonical monotone operator associated with the given arbitrary operator and hence they are also covered by Tarski's theorem.
In this chapter we only consider the type systems for inductive types, the following chapters introduce the term systems together with notions of β-reduction for μ-types which represent iteration and (full primitive) recursion.

[1] In the context of inductive definitions it is customary to speak of monotone operators instead of monotone (endo-)functions, see e. g. [Acz77].

[2] In [Tar55] it is even shown that the set of fixed points is a complete lattice and in particular there is also a greatest fixed point. Those greatest fixed points are the natural semantics for coinductive types. (Note that "coinductive" does not refer to the complement of an inductively defined set as in [Mos74, p. 17].) Note also that the existence of fixed points of every monotone operator even characterizes the completeness property of lattices [Dav55].

3.1 Inductive types

The type system of all of the extensions by inductive types which we will consider (unless some subsystem is studied) is defined by adding the following clause to the inductive definition of Types for eF:

(μ) If $\alpha \in \mathsf{Typevars}$ and $\rho \in \mathsf{Types}$, then $\mu\alpha\rho \in \mathsf{Types}$.

As was the case for i and $\forall$, the variable α in $\mu\alpha\rho$ is considered to be bound by μ. Hence, the renaming convention for bound type variables in types will be extended accordingly. μ is assumed to syntactically bind stronger than $\to$, $\times$ and $+$, as was done for $\forall$ (and also for i).
The definition of $\mathsf{FV}(\rho)$ for system eF is extended by the following clause:

(μ) $\mathsf{FV}(\mu\alpha\rho) := \mathsf{FV}(\rho) \setminus \{\alpha\}$.

The definition of $\rho[\vec{\alpha} := \vec{\sigma}]$ is extended by the following clause:

(μ) $(\mu\alpha\rho)[\vec{\alpha} := \vec{\sigma}] := \mu\alpha.\rho[\vec{\alpha} := \vec{\sigma}]$, where we assume that $\alpha \notin \vec{\alpha} \cup \mathsf{FV}(\vec{\sigma})$.

Again μ and $\forall$ are only different because of their different names and therefore the statements in section 2.1.1 remain true in this extension.
The intuitive meaning of $\mu\alpha\rho$ is the least fixed point of $\sigma \mapsto \rho[\alpha := \sigma]$ the existence of which is guaranteed for monotone $\lambda\alpha\rho$.
Some examples are in order:

- $\mathsf{nat} := \mu\alpha.1 + \alpha$ is the type of natural numbers.
- $\mathsf{list}(\rho) := \mu\alpha.1 + \rho \times \alpha$ for $\alpha \notin \mathsf{FV}(\rho)$ is the type of finite lists with elements of type ρ.
- $\mathsf{tree}(\rho) := \mu\alpha.1 + (\rho \to \alpha)$ for $\alpha \notin \mathsf{FV}(\rho)$ is the type of ρ-branching well-founded trees.
- $\mathsf{tree}'(\rho) := \mu\alpha.1 + \alpha + (\rho \to \alpha)$ for $\alpha \notin \mathsf{FV}(\rho)$ is the type of ρ-branching well-founded trees with successor.
- $\mathsf{tree} := \mathsf{tree}(\mathsf{nat})$ is a type representing Kleene's $\mathcal{O}$.
- $\mathsf{fin} := \mu\alpha.\mathsf{list}(\alpha)$ is the type of finitely branching well-founded trees.
- $\mathsf{cont}(\rho) := \mu\alpha.1 + ((\alpha \to \rho) \to \rho)$ for $\alpha \notin \mathsf{FV}(\rho)$ is the type of "continuations for calculating results of type ρ" (which are used in an unpublished manuscript written by Martin Hofmann).
- $\mathsf{Tree} := \mu\alpha.1 + (\mathsf{tree}(\alpha) \to \alpha)$ (which was first proposed by Ulrich Berger) is a type of trees with autonomously generated branching degree. It would be interesting to see in which way this type is related to autonomously iterated inductive definitions (see the theory AUT(ID) in [BFPS81]).
- $\mu\alpha.\alpha \to 1$, which is trivially monotone but not positive (for its use see appendix B).
- $\mathsf{bizarre}(\rho) := \mu\alpha.1 + ((((\alpha \to \rho) \to \alpha) \to \alpha) \to \alpha)$ for $\alpha \notin \mathsf{FV}(\rho)$ (an invention by Ulrich Berger of a non-trivially monotone non-positive inductive type discussed in section 4.1.2).
- $\mathsf{pos}(\mu\alpha\rho) := \mu\alpha\forall\beta.(\alpha \to \beta) \to \rho[\alpha := \beta]$ for $\beta \notin \{\alpha\} \cup \mathsf{FV}(\rho)$ is the "positivization" of $\mu\alpha\rho$ playing a prominent role in section 6.3.3.

3.2 Strictly positive and positive types

In the following we will define subsets of Types where we restrict the use of the μ-rule. Inductively define the set $\mathsf{SPosTypes}$ of strictly positive types (as a subset of Types) and simultaneously for every $\rho \in \mathsf{SPosTypes}$ the set $\mathsf{SPos}(\rho)$ of type variables which only occur free at strictly positive positions in ρ:

(V) $\alpha \in \mathsf{SPosTypes}$ and $\mathsf{SPos}(\alpha) := \mathsf{Typevars}$.

($\to$) If $\rho \in \mathsf{SPosTypes}$ and $\sigma \in \mathsf{SPosTypes}$, then $\rho \to \sigma \in \mathsf{SPosTypes}$ and $\mathsf{SPos}(\rho \to \sigma) := \mathsf{SPos}(\sigma) \setminus \mathsf{FV}(\rho)$.

($\forall$) If $\rho \in \mathsf{SPosTypes}$, then $\forall\alpha\rho \in \mathsf{SPosTypes}$ and $\mathsf{SPos}(\forall\alpha\rho) := \mathsf{SPos}(\rho) \cup \{\alpha\}$.

(0) $0 \in \mathsf{SPosTypes}$ and $\mathsf{SPos}(0) := \mathsf{Typevars}$.

(1) $1 \in \mathsf{SPosTypes}$ and $\mathsf{SPos}(1) := \mathsf{Typevars}$.

($\times$) If $\rho \in \mathsf{SPosTypes}$ and $\sigma \in \mathsf{SPosTypes}$, then $\rho \times \sigma \in \mathsf{SPosTypes}$ and $\mathsf{SPos}(\rho \times \sigma) := \mathsf{SPos}(\rho) \cap \mathsf{SPos}(\sigma)$.

($+$) If $\rho \in \mathsf{SPosTypes}$ and $\sigma \in \mathsf{SPosTypes}$, then $\rho + \sigma \in \mathsf{SPosTypes}$ and $\mathsf{SPos}(\rho + \sigma) := \mathsf{SPos}(\rho) \cap \mathsf{SPos}(\sigma)$.

(μ) If $\rho \in \mathsf{SPosTypes}$ and $\alpha \in \mathsf{SPos}(\rho)$, then $\mu\alpha\rho \in \mathsf{SPosTypes}$ and $\mathsf{SPos}(\mu\alpha\rho) := \mathsf{SPos}(\rho)$.

This definition is compatible with the renaming convention: This is implied by the following lemma which in a more careful presentation would be proved simultaneously with the justification of the renaming convention.

Lemma 3.1 Let $\rho, \vec{\sigma} \in \mathsf{SPosTypes}$. Then $\rho[\vec{\alpha} := \vec{\sigma}] \in \mathsf{SPosTypes}$ and $\mathsf{SPos}(\rho) \setminus \mathsf{FV}(\vec{\sigma}) \subseteq \mathsf{SPos}(\rho[\vec{\alpha} := \vec{\sigma}])$. Moreover, if $\beta \notin \mathsf{FV}(\rho)$ then $\alpha \in \mathsf{SPos}(\rho)$ implies $\beta \in \mathsf{SPos}(\rho[\alpha := \beta])$.

Proof Induction on $\rho \in \mathsf{SPosTypes}$. For the case (V) note that every type variable which does not occur free in ρ is put in $\mathsf{SPos}(\rho)$ (e. g. $\mathsf{SPos}(\mu\alpha\rho) = \mathsf{SPos}(\rho) \cup \{\alpha\}$). In the case ($\mu$) the induction hypothesis for the second claim is needed to prove the first claim. □

It follows that if $\mu\alpha\rho \in \mathsf{SPosTypes}$ and $\beta \notin \mathsf{FV}(\rho)$, then $\mu\beta.\rho[\alpha := \beta] \in \mathsf{SPosTypes}$. But $\mu\beta.\rho[\alpha := \beta] = \mu\alpha\rho$. However, this part of the lemma is only intended to show compatibility with the renaming convention.
Let us look at the examples of inductive types:

- $\mathsf{nat} \in \mathsf{SPosTypes}$.
- If $\rho \in \mathsf{SPosTypes}$, then $\mathsf{list}(\rho) \in \mathsf{SPosTypes}$. In this case $\mathsf{SPos}(\mathsf{list}(\rho)) = \mathsf{SPos}(\rho)$.
- If $\rho \in \mathsf{SPosTypes}$, then $\mathsf{tree}(\rho) \in \mathsf{SPosTypes}$. In this case $\mathsf{SPos}(\mathsf{tree}(\rho)) \cap \mathsf{FV}(\rho) = \emptyset$. The same holds for $\mathsf{tree}'(\rho)$.
- tree and fin are strictly positive types.
- $\mathsf{cont}(\rho)$, Tree, $\mathsf{bizarre}$ and $\mathsf{pos}(\mu\alpha\rho)$ are never in $\mathsf{SPosTypes}$.

It is common to view types of the form $\mu\alpha\rho \in \mathsf{SPosTypes}$ as the types of well-founded trees (at least if the rule ($\forall$) is left out). But trees in this sense are not the only structures definable by induction. Our understanding of an inductive definition rests on some monotone operator described by the definition. The elements of a deterministic inductive definition may be identified with the proof trees which justify their elementhood (see e. g. [Acz77, p. 748]). In this manner we may view all of our μ-types as types of generalized well-founded trees. However, we leave the realm of predicativity if we go beyond strictly positive types. Of course, we do this anyway by basing our systems on system F, but one could try to produce predicative versions of them. However, there is no general agreement as to where the "border of predicativity" is (see [Str99] on "metapredicativity"). Clearly, system F is impredicative. Strictly positive inductive definitions (without $\forall$) are mostly seen as predicative. However, this work does not aim at determining the exact proof-theoretic strength of theories but investigates reduction behaviour by unrestricted use of Tarski's fixed-point theorem (which makes use of a Π^1_1-comprehension) and by embeddings using the expressive power of system F. Hence, we simply regard strictly positive types as an instance of monotone inductive types where monotonicity already may be checked syntactically by looking at the type.
And if we ask ourselves the question which types may also syntactically be checked for monotonicity, we are directly led to the positive inductive types. Because of the contravariance of $\to$ (seen as a functor) in the first argument we have to consider syntactically isotone and syntactically antitone types (seen as functions of a type variable) simultaneously:
Inductively define the set $\mathsf{PosTypes}$ of positive types (as a subset of Types) and simultaneously for every $\rho \in \mathsf{PosTypes}$ the set $\mathsf{Pos}(\rho)$ of type variables which only occur free at positive positions in ρ and the set $\mathsf{Neg}(\rho)$ of type variables which only occur free at negative positions in ρ:

(V) $\alpha \in \mathsf{PosTypes}$ and $\mathsf{Pos}(\alpha) := \mathsf{Typevars}$ and $\mathsf{Neg}(\alpha) := \mathsf{Typevars} \setminus \{\alpha\}$.

($\to$) If $\rho \in \mathsf{PosTypes}$ and $\sigma \in \mathsf{PosTypes}$, then $\rho \to \sigma \in \mathsf{PosTypes}$ and $\mathsf{Pos}(\rho \to \sigma) := \mathsf{Neg}(\rho) \cap \mathsf{Pos}(\sigma)$ and $\mathsf{Neg}(\rho \to \sigma) := \mathsf{Pos}(\rho) \cap \mathsf{Neg}(\sigma)$.

($\forall$) If $\rho \in \mathsf{PosTypes}$, then $\forall\alpha\rho \in \mathsf{PosTypes}$. $\mathsf{Pos}(\forall\alpha\rho) := \mathsf{Pos}(\rho) \cup \{\alpha\}$ and $\mathsf{Neg}(\forall\alpha\rho) := \mathsf{Neg}(\rho) \cup \{\alpha\}$.

(0) $0 \in \mathsf{PosTypes}$ and $\mathsf{Pos}(0) := \mathsf{Typevars}$ and $\mathsf{Neg}(0) := \mathsf{Typevars}$.

(1) $1 \in \mathsf{PosTypes}$ and $\mathsf{Pos}(1) := \mathsf{Typevars}$ and $\mathsf{Neg}(1) := \mathsf{Typevars}$.

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

($+$) If $\rho \in \mathsf{PosTypes}$ and $\sigma \in \mathsf{PosTypes}$, then $\rho + \sigma \in \mathsf{PosTypes}$ and $\mathsf{Pos}(\rho + \sigma) := \mathsf{Pos}(\rho) \cap \mathsf{Pos}(\sigma)$ and $\mathsf{Neg}(\rho + \sigma) := \mathsf{Neg}(\rho) \cap \mathsf{Neg}(\sigma)$.

(μ) If $\rho \in \mathsf{PosTypes}$ and $\alpha \in \mathsf{Pos}(\rho)$, then $\mu\alpha\rho \in \mathsf{PosTypes}$ and $\mathsf{Pos}(\mu\alpha\rho) := \mathsf{Pos}(\rho)$ and $\mathsf{Neg}(\mu\alpha\rho) := \mathsf{Neg}(\rho) \cup \{\alpha\}$.

The definition is again compatible with the renaming convention: This is implied by the following lemma which also in a more careful presentation would be proved simultaneously with the justification of the renaming convention.

Lemma 3.2 Let $\rho, \vec{\sigma} \in \mathsf{PosTypes}$. Then $\rho[\vec{\alpha} := \vec{\sigma}] \in \mathsf{PosTypes}$ and

$$\mathsf{Pos}(\rho) \setminus \mathsf{FV}(\vec{\sigma}) \subseteq \mathsf{Pos}(\rho[\vec{\alpha} := \vec{\sigma}]) \text{ and } \mathsf{Neg}(\rho) \setminus \mathsf{FV}(\vec{\sigma}) \subseteq \mathsf{Neg}(\rho[\vec{\alpha} := \vec{\sigma}]).$$

Moreover, if $\beta \notin \mathsf{FV}(\rho)$ then $\alpha \in \mathsf{Pos}(\rho)$ implies $\beta \in \mathsf{Pos}(\rho[\alpha := \beta])$ and $\alpha \in \mathsf{Neg}(\rho)$ implies $\beta \in \mathsf{Neg}(\rho[\alpha := \beta])$.

Proof Induction on $\rho \in \mathsf{PosTypes}$. For the case (V) note that every type variable which does not occur free in ρ is put in $\mathsf{Pos}(\rho)$ and $\mathsf{Neg}(\rho)$ (e.g. $\mathsf{Pos}(\mu\alpha\rho) = \mathsf{Pos}(\rho) \cup \{\alpha\}$). In the case ($\mu$) the induction hypothesis for the second and third claim is needed to prove the first claim. $\square$

It follows that if $\mu\alpha\rho \in \mathsf{PosTypes}$ and $\beta \notin \mathsf{FV}(\rho)$, then $\mu\beta.\rho[\alpha := \beta] \in \mathsf{PosTypes}$. We knew that before as $\mu\beta.\rho[\alpha := \beta] = \mu\alpha\rho$. However, this part of the lemma is again only intended to show compatibility with the renaming convention.

Lemma 3.3 For $\rho \in \mathsf{PosTypes}$, $\mathsf{Pos}(\rho) \cap \mathsf{Neg}(\rho) \cap \mathsf{FV}(\rho) = \emptyset$.

Proof Induction on ρ. $\square$

Lemma 3.4 $\mathsf{SPosTypes} \subset \mathsf{PosTypes}$ and for every $\rho \in \mathsf{SPosTypes}$, we have that $\mathsf{SPos}(\rho) \subseteq \mathsf{Pos}(\rho)$.

Proof Induction on the definition of $\mathsf{SPosTypes}$ and $\mathsf{SPos}(\rho)$. $\square$

We have that $\mathsf{PosTypes} \neq \mathsf{SPosTypes}$, as e.g. $\mu\alpha.(\alpha \to 0) \to 0 \in \mathsf{PosTypes} \setminus \mathsf{SPosTypes}$.
Concerning the examples of inductive types:

- If $\rho \in \mathsf{PosTypes}$, then $\mathsf{list}(\rho) \in \mathsf{PosTypes}$. In this case $\mathsf{Pos}(\mathsf{list}(\rho)) = \mathsf{Pos}(\rho)$ and $\mathsf{Neg}(\mathsf{list}(\rho)) = \mathsf{Neg}(\rho)$.
- If $\rho \in \mathsf{PosTypes}$, then $\mathsf{tree}(\rho) \in \mathsf{PosTypes}$. In this case we have the reverse situation, namely $\mathsf{Pos}(\mathsf{tree}(\rho)) = \mathsf{Neg}(\rho)$ and $\mathsf{Neg}(\mathsf{tree}(\rho)) = \mathsf{Pos}(\rho)$. (The same holds for $\mathsf{tree}'(\rho)$.)
- If $\rho \in \mathsf{PosTypes}$, then $\mathsf{cont}(\rho) \in \mathsf{PosTypes}$. In this case $\mathsf{Pos}(\mathsf{cont}(\rho)) \cap \mathsf{FV}(\rho) = \emptyset$ and $\mathsf{Neg}(\mathsf{cont}(\rho)) \cap \mathsf{FV}(\rho) = \emptyset$.
- $\mathsf{Tree} \in \mathsf{PosTypes}$ and $\mathsf{bizarre} \notin \mathsf{PosTypes}$.
- If $\rho \in \mathsf{PosTypes}$, then $\mathsf{pos}(\mu\alpha\rho) \in \mathsf{PosTypes}$. In this case $\mathsf{Pos}(\mathsf{pos}(\mu\alpha\rho)) = \mathsf{Pos}(\mu\alpha\rho)$ and $\mathsf{Neg}(\mathsf{pos}(\mu\alpha\rho)) = \mathsf{Neg}(\mu\alpha\rho)$. The crucial point is that we did not assume $\mu\alpha\rho \in \mathsf{PosTypes}$, i.e., $\alpha \in \mathsf{Pos}(\rho)$ is not required.

3.3 Height and depth for inductive types

Abstracted types are objects of the form $\lambda\vec{\alpha}\rho$, where $\vec{\alpha}$ is a finite list of type variables and ρ is a type. The type variables $\vec{\alpha}$ are assumed to be bound in the abstracted type (the priority being from the right to the left), and we will extend the renaming convention to abstracted types. If $\vec{\alpha}$ is the list $(\alpha_1, \alpha_2, \ldots, \alpha_n)$, we will also write $\lambda\alpha_1\lambda\alpha_2\ldots\lambda\alpha_n\rho$ for $\lambda\vec{\alpha}\rho$.
Define $(\lambda\vec{\alpha}\rho)[\vec{\gamma} := \vec{\sigma}] := \lambda\vec{\alpha}.\rho[\vec{\gamma} := \vec{\sigma}]$, where we assume that $\vec{\alpha} \cap (\vec{\gamma} \cup \mathsf{FV}(\vec{\sigma})) = \emptyset$.
For every abstracted type $\lambda\vec{\alpha}\rho$ define its height $\mathsf{h}(\lambda\vec{\alpha}\rho) \in \mathbb{N}$ by recursion on ρ as follows:

(triv) $\mathsf{h}(\lambda\vec{\alpha}\rho) := 0$ if $\vec{\alpha} \cap \mathsf{FV}(\rho) = \emptyset$ (e.g. $\rho = 0$ or 1). The following rules will be under the proviso "otherwise".

(V) $\mathsf{h}(\lambda\vec{\alpha}\alpha) := 0$ (the proviso implies $\alpha \in \vec{\alpha}$)

($\circ$) $\mathsf{h}(\lambda\vec{\alpha}.\rho \circ \sigma) := 1 + \max(\mathsf{h}(\lambda\vec{\alpha}\rho), \mathsf{h}(\lambda\vec{\alpha}\sigma))$ for $\circ \in \{\to, \times, +\}$

(∇) $\mathsf{h}(\lambda\vec{\alpha}\nabla\alpha\rho) := 1 + \mathsf{h}(\lambda(\vec{\alpha}, \alpha)\rho)$ for $\nabla \in \{\forall, \mu\}$

This definition is compatible with the renaming convention (for abstracted types).

Lemma 3.5
1. $\mathsf{h}((\lambda\vec{\alpha}\rho)[\vec{\gamma} := \vec{\sigma}]) = \mathsf{h}(\lambda\vec{\alpha}\rho)$.
2. $\mathsf{h}(\lambda\vec{\alpha}\rho) \geq \mathsf{h}(\lambda\vec{\alpha}'\rho)$ if $\vec{\alpha}' \subseteq \vec{\alpha}$.
3. $\mathsf{h}(\lambda\vec{\alpha}\nabla\gamma\rho) > \mathsf{h}((\lambda\gamma\rho)[\vec{\alpha} := \vec{\sigma}])$ if $\vec{\alpha} \cap \mathsf{FV}(\nabla\gamma\rho) \neq \emptyset$ (for $\nabla \in \{\forall, \mu\}$).
4. $\mathsf{h}(\lambda\vec{\alpha}\nabla\gamma\rho) > \mathsf{h}((\lambda\vec{\alpha}\rho)[\gamma := \sigma])$ if $\vec{\alpha} \cap \mathsf{FV}(\nabla\gamma\rho) \neq \emptyset$ (for $\nabla \in \{\forall, \mu\}$).

Proof The first two statements are easily proved by induction on ρ. 3.: $\mathsf{h}(\lambda\vec{\alpha}\nabla\gamma\rho) \overset{\text{Def.}}{=} 1 + \mathsf{h}(\lambda(\vec{\alpha}, \gamma)\rho) \geq 1 + \mathsf{h}(\lambda\gamma\rho) \overset{1.}{=} 1 + \mathsf{h}((\lambda\gamma\rho)[\vec{\alpha} := \vec{\sigma}])$, where the $\geq$ uses 2. The fourth statement is proved similarly. $\square$

It is possible to show that $\mathsf{h}(\lambda\vec{\alpha}\rho)$ is the height (in the ordinary sense given by only counting connectives and $\forall$ and μ) of a maximally general generalization of $\lambda\vec{\alpha}\rho$. This is not done here because of the unpleasant work of defining this concept properly. Nevertheless, we do not need this characterization for our purposes.
In most cases $\mathsf{h}(\lambda\vec{\alpha}\rho)$ will be used for abstracted types with a list $\vec{\alpha}$ of length 1. We will now study another measure d of height for abstracted types of this kind where the definition does not use the "detour" through multiply abstracted types. Intuitively, $\mathsf{d}(\lambda\alpha\rho)$ will be the maximal depth[3] of a free occurrence of α in ρ.
For every abstracted type of the form $\lambda\alpha\rho$ define its depth $\mathsf{d}(\lambda\alpha\rho) \in \mathbb{N}$ by recursion on ρ and simultaneously prove that $\mathsf{d}(\lambda\alpha\rho[\vec{\alpha} := \vec{\sigma}]) = \mathsf{d}(\lambda\alpha\rho)$ for $\alpha \notin \vec{\alpha} \cup \mathsf{FV}(\vec{\sigma})$ as follows:

(triv) $\mathsf{d}(\lambda\alpha\rho) := 0$ if $\alpha \notin \mathsf{FV}(\rho)$ (e.g. $\rho = 0$ or 1). The following rules will be under the proviso "otherwise". By Corollary 2.4, we have the same proviso for $\mathsf{d}(\lambda\alpha\rho[\vec{\alpha} := \vec{\sigma}])$.

(V) $\mathsf{d}(\lambda\alpha\alpha) := 0$. Proof trivial.

($\circ$) $\mathsf{d}(\lambda\alpha.\rho \circ \sigma) := 1 + \max(\mathsf{d}(\lambda\alpha\rho), \mathsf{d}(\lambda\alpha\sigma))$ for $\circ \in \{\to, \times, +\}$. Proof obvious.

(∇) $\mathsf{d}(\lambda\alpha\nabla\gamma\rho) := 1 + \mathsf{d}(\lambda\alpha\rho)$ for $\nabla \in \{\forall, \mu\}$. By induction hypothesis this definition is compatible with a renaming of γ. The proof of the claim for $\lambda\alpha\nabla\gamma\rho$ is easy.

This definition is compatible with the renaming convention (for abstracted types) because of the simultaneously proved claim.
For ease of reference we formulate

Lemma 3.6
1. $\mathsf{d}((\lambda\alpha\rho)[\vec{\alpha} := \vec{\sigma}]) = \mathsf{d}(\lambda\alpha\rho)$.
2. $\mathsf{d}(\lambda\alpha\nabla\gamma\rho) = 1 + \mathsf{d}((\lambda\alpha\rho)[\gamma := \sigma])$ if $\alpha \in \mathsf{FV}(\nabla\gamma\rho)$ (for $\nabla \in \{\forall, \mu\}$).

Proof 1.: Proved in the definition (we may assume that $\alpha \notin \vec{\alpha} \cup \mathsf{FV}(\vec{\sigma})$). 2.: By definition and by 1. $\square$

Lemma 3.7 $\mathsf{d}(\lambda\alpha\rho) \leq \mathsf{h}(\lambda\alpha\rho)$.

Proof Induction on ρ using Lemma 3.5 (2.) for the cases ($\forall$) and (μ). $\square$

Equality does not hold: $\mathsf{h}(\lambda\alpha\forall\gamma.(\gamma \to \gamma) \to \alpha) = 3$ and $\mathsf{d}(\lambda\alpha\forall\gamma.(\gamma \to \gamma) \to \alpha) = 2$.

[3] This concept seems more intuitive than the one lying behind the definition of h. However, it is more problematic with respect to our variable convention because variables become unbound in the recursive call. Moreover, some proofs do not go through with d as induction measure instead of h.

3.4 Extension by the existential quantifier

The term systems with suffix $+\mathsf{ex}$ in the name will be based on the further extension by existential types. Hence, the type system they will be based on is the system of inductive types extended by the clause $(\exists)$ exactly in the same way as it was done in section 2.4.1 where the clause was added to the type system of eF.
Extend the inductive definition of $\mathsf{SPosTypes}$ and the simultaneous recursive definition of $\mathsf{SPos}(\rho)$ for every $\rho \in \mathsf{SPosTypes}$ by the following clause:

$(\exists)$ If $\rho \in \mathsf{SPosTypes}$, then $\exists\alpha\rho \in \mathsf{SPosTypes}$ and $\mathsf{SPos}(\exists\alpha\rho) := \mathsf{SPos}(\rho) \cup \{\alpha\}$.

The statement of Lemma 3.1 is valid also for this extension and implies compatibility of the definition with the renaming convention for bound variables.
Extend the inductive definition of $\mathsf{PosTypes}$ and the simultaneous recursive definition of $\mathsf{Pos}(\rho)$ and $\mathsf{Neg}(\rho)$ for every $\rho \in \mathsf{PosTypes}$ by the following clause:

$(\exists)$ If $\rho \in \mathsf{PosTypes}$, then $\exists\alpha\rho \in \mathsf{PosTypes}$. $\mathsf{Pos}(\exists\alpha\rho) := \mathsf{Pos}(\rho) \cup \{\alpha\}$. $\mathsf{Neg}(\exists\alpha\rho) := \mathsf{Neg}(\rho) \cup \{\alpha\}$.

The statement of Lemma 3.2 holds again for this extension and implies compatibility with the variable convention. Also the statement of Lemma 3.4 holds.
All the extensions to the proofs are immediate because $(\exists)$ differs from $(\forall)$ only in the name of the binder. For this reason the extension of the definition of h and d and Lemma 3.5, Lemma 3.6 and Lemma 3.7 are also trivial: Simply allow $\nabla \in \{\forall, \mu, \exists\}$.
Recall the embedding of $\mathsf{eF}{+}\mathsf{ex}$ in eF. The only clause of interest in the definition of ρ' was

$(\exists)$ $(\exists\alpha\rho)' := \forall\beta.(\forall\alpha.\rho' \to \beta) \to \beta$ for $\beta \notin \{\alpha\} \cup \mathsf{FV}(\rho)$.

This encoding of the existential quantifier is unproblematic with respect to positive types but leads out of the strictly positive types. To be more precise: If $\exists\alpha\rho \in \mathsf{PosTypes}$ and $\gamma \in \mathsf{Pos}(\exists\alpha\rho)$, then $\forall\beta.(\forall\alpha.\rho \to \beta) \to \beta \in \mathsf{PosTypes}$ and $\gamma \in \mathsf{Pos}(\forall\beta.(\forall\alpha.\rho \to \beta) \to \beta)$. On the other hand, $\exists\alpha\gamma \in \mathsf{SPosTypes}$, $\gamma \in \mathsf{SPos}(\exists\alpha\gamma)$ and $\forall\beta.(\forall\alpha.\gamma \to \beta) \to \beta \in \mathsf{SPosTypes}$, but we have $\gamma \notin \mathsf{SPos}(\forall\beta.(\forall\alpha.\gamma \to \beta) \to \beta)$ (if $\gamma \notin \{\alpha, \beta\}$). Hence, $\mu\gamma\exists\alpha\gamma \in \mathsf{SPosTypes}$ and for $\gamma \notin \{\alpha, \beta\}$ we have $\mu\gamma\forall\beta.(\forall\alpha.\gamma \to \beta) \to \beta \notin \mathsf{SPosTypes}$.
As a further example of inductive types consider

- $\mathsf{spos}(\mu\alpha\rho) := \mu\alpha\exists\beta.(\beta \to \alpha) \times \rho[\alpha := \beta]$ for $\beta \notin \{\alpha\} \cup \mathsf{FV}(\rho)$. This type is the "strict positivization" of $\mu\alpha\rho$ which will be essential in section 6.2.3.

We note that if $\rho \in \mathsf{SPosTypes}$, then $\mathsf{spos}(\mu\alpha\rho) \in \mathsf{SPosTypes}$ and $\mathsf{SPos}(\mathsf{spos}(\mu\alpha\rho)) = \mathsf{SPos}(\mu\alpha\rho)$. Moreover, if $\rho \in \mathsf{PosTypes}$, then $\mathsf{spos}(\mu\alpha\rho) \in \mathsf{PosTypes}$ and $\mathsf{Pos}(\mathsf{spos}(\mu\alpha\rho)) = \mathsf{Pos}(\mu\alpha\rho)$ and $\mathsf{Neg}(\mathsf{spos}(\mu\alpha\rho)) = \mathsf{Neg}(\mu\alpha\rho)$.

3.5 Restriction to non-interleaving positive inductive types

The term rewrite systems whose names start with NI as NIPIT, NISPIT and $\mathsf{NISPIT}{+}\mathsf{ex}$, but also MePIT and $\mathsf{coMePIT}$, will be based on type systems where nested applications of the μ-rule are only allowed as long as they do not interleave, i. e., types of the form $\mu\alpha\rho$ are ruled out if ρ has a subexpression of the form $\mu\beta\sigma$ where α occurs free. More formally: Inductively define the set $\mathsf{NIPosTypes}$ of non-interleaving 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: Repeat the definition of $\mathsf{PosTypes}$, Pos and Neg in the cases (V), $(\to)$, $(\forall)$, (0), (1), $(\times)$ and $(+)$ by replacing the words accordingly.
Case (μ): If $\rho \in \mathsf{NIPosTypes}$ and $\alpha \in \mathsf{NIPos}(\rho)$, then $\mu\alpha\rho \in \mathsf{NIPosTypes}$ and $\mathsf{NIPos}(\mu\alpha\rho) := \mathsf{NIPos}(\rho) \setminus \mathsf{FV}(\mu\alpha\rho)$ and $\mathsf{NINeg}(\mu\alpha\rho) := (\mathsf{NINeg}(\rho) \setminus \mathsf{FV}(\rho)) \cup \{\alpha\}$.

Lemma 3.8 $\mathsf{NIPosTypes} \subset \mathsf{PosTypes}$ and for every $\rho \in \mathsf{NIPosTypes}$, we have $\mathsf{NIPos}(\rho) \subseteq \mathsf{Pos}(\rho)$ and $\mathsf{NINeg}(\rho) \subseteq \mathsf{Neg}(\rho)$.

Proof Induction on the definition. □

We have that $\mathsf{NIPosTypes} \neq \mathsf{PosTypes}$, as e. g. $\mathsf{fin} = \mu\alpha\mu\beta.1 + \alpha \times \beta \in \mathsf{PosTypes} \setminus \mathsf{NIPosTypes}$.

Lemma 3.9 If $\rho \in \mathsf{NIPosTypes}$, then $\mathsf{Typevars} \setminus \mathsf{FV}(\rho) \subseteq \mathsf{NIPos}(\rho) \cap \mathsf{NINeg}(\rho)$.

Proof Induction on ρ. □

Lemma 3.10 The statement of Lemma 3.2 is also valid if the words $\mathsf{PosTypes}$, Pos and Neg are replaced by $\mathsf{NIPosTypes}$, NIPos and NINeg, respectively.

Proof Induction on $\rho \in \mathsf{NIPosTypes}$. Evidently, there are only changes in the case (μ): The proof of the first statement in Lemma 3.2 is refined by reference to Lemma 2.3, the proof of the last statement becomes very easy because w.l.o.g. $\alpha \notin \mathsf{FV}(\rho)$. □

Define the set $\mathsf{NISPosTypes}$ and for every $\rho \in \mathsf{NISPosTypes}$ the set $\mathsf{NISPos}(\rho)$ of type variables which only occur free at strictly positive positions in ρ and are not bound by an application of the μ-rule in the same way: Take the definition of $\mathsf{SPosTypes}$ and SPos and replace the words accordingly. The case (μ) becomes:
If $\rho \in \mathsf{NISPosTypes}$ and $\alpha \in \mathsf{NISPos}(\rho)$, then $\mu\alpha\rho \in \mathsf{NISPosTypes}$ and $\mathsf{NISPos}(\mu\alpha\rho) := \mathsf{NISPos}(\rho) \setminus \mathsf{FV}(\mu\alpha\rho)$.

Lemma 3.11 If $\rho \in \mathsf{NISPosTypes}$, then $\mathsf{Typevars} \setminus \mathsf{FV}(\rho) \subseteq \mathsf{NISPos}(\rho)$.

Proof Induction on ρ. □

It quite obvious that $\mathsf{NISPosTypes}$ stands in the same relation to $\mathsf{NIPosTypes}$ as $\mathsf{SPosTypes}$ stands to $\mathsf{PosTypes}$ and consequently $\mathsf{NISPosTypes}$ is simply the intersection of $\mathsf{SPosTypes}$ and $\mathsf{NIPosTypes}$.
The extension by the existential quantifier is again trivial (because $\forall$ and $\exists$ are treated exactly the same).
Let us review the examples of inductive types:

- $\mathsf{nat} \in \mathsf{NISPosTypes}$.
- If $\rho \in \mathsf{NISPosTypes}$, then $\mathsf{list}(\rho) \in \mathsf{NISPosTypes}$ and $\mathsf{NISPos}(\mathsf{list}(\rho)) \cap \mathsf{FV}(\rho) = \emptyset$.
  If $\rho \in \mathsf{NIPosTypes}$, then $\mathsf{list}(\rho) \in \mathsf{NIPosTypes}$ and $\mathsf{NIPos}(\mathsf{list}(\rho)) \cap \mathsf{FV}(\rho) = \emptyset$ and also $\mathsf{NINeg}(\mathsf{list}(\rho)) \cap \mathsf{FV}(\rho) = \emptyset$. The same holds for $\mathsf{tree}(\rho)$ and $\mathsf{tree}'(\rho)$.
- $\mathsf{tree} \in \mathsf{NISPosTypes}$ and $\mathsf{fin} \notin \mathsf{NIPosTypes}$, because tree is only nested but fin is interleaved.
- If $\rho \in \mathsf{NIPosTypes}$, then $\mathsf{cont}(\rho) \in \mathsf{NIPosTypes}$.
- $\mathsf{Tree} \notin \mathsf{NIPosTypes}$.
- If $\rho \in \mathsf{NIPosTypes}$, then $\mathsf{pos}(\mu\alpha\rho) \in \mathsf{NIPosTypes}$ due to Lemma 3.9.
- If $\rho \in \mathsf{NISPosTypes}$, then $\mathsf{spos}(\mu\alpha\rho) \in \mathsf{NISPosTypes}$ due to Lemma 3.11. If $\rho \in \mathsf{NIPosTypes}$, then $\mathsf{spos}(\mu\alpha\rho) \in \mathsf{NIPosTypes}$ (again due to Lemma 3.9).

Chapter 4

Systems with monotone inductive types

Let $(U, \subseteq)$ be a complete lattice of sets and Φ a monotone operator on it. In the introduction to the preceding chapter we stated the following corollary to Tarski's fixed-point theorem:

(lfp-E) If $x \in \mathsf{lfp}(\Phi)$ and $\Phi(M) \subseteq M$, then $x \in M$.

In our naïve realizability interpretation we interpret Φ by some $\lambda\alpha\rho$, $\mathsf{lfp}(\Phi)$ by $\mu\alpha\rho$ and M by some type σ. The truth of $x \in \mathsf{lfp}(\Phi)$ is modelled by some term r of type $\mu\alpha\rho$ and the truth of $\Phi(M) \subseteq M$ by some term s of type $\rho[\alpha := \sigma] \to \sigma$. We conclude the truth of $x \in M$, which hence is to be modelled by a term of type σ constructed out of r and s. This motivates the following rule[1]
of μ-elimination for EMIT (to be defined in the next section)

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

Let us now introduce the concept of "extended induction" (which will turn out to be the counterpart of full primitive recursion).
Let $(U, \leq)$ be a complete lattice and Φ a monotone operator on it and $M \in U$. We apply the minimality of $\mathsf{lfp}(\Phi)$ to $M' := \mathsf{lfp}(\Phi) \wedge M$.[2] Therefore $\mathsf{lfp}(\Phi) \leq M'$, if $\Phi(M') \leq M'$. Let $\Phi(M') \leq M$. Because $M' \leq \mathsf{lfp}(\Phi)$ and Φ is monotone, also $\Phi(M') \leq \Phi(\mathsf{lfp}(\Phi))$. Because $\mathsf{lfp}(\Phi)$ is a pre-fixed-point of Φ (it is even a fixed point) and due to transitivity $\Phi(M') \leq \mathsf{lfp}(\Phi)$. Hence, $\Phi(M') \leq M'$. Using the trivial fact that $\mathsf{lfp}(\Phi) \leq M'$ implies $\mathsf{lfp}(\Phi) \leq M$, we conclude

(lfp-E$^+$) $\Phi(\mathsf{lfp}(\Phi) \wedge M) \leq M \Rightarrow \mathsf{lfp}(\Phi) \leq M$.

Let now $(U, \subseteq)$ be even a complete lattice of sets and Φ be monotone. The preceding rule reads

(lfp-E$^+$) If $x \in \mathsf{lfp}(\Phi)$ and $\Phi(\mathsf{lfp}(\Phi) \wedge M) \subseteq M$, then $x \in M$.

This shall be called extended induction on $\mathsf{lfp}(\Phi)$. As for (lfp-E) this directly motivates the following second rule[3] of μ-elimination for EMIT:

[1]Note that it would be more traditional to assume an iterator constant $I_{\mu\alpha\rho,\sigma}$ of type $(\rho[\alpha := \sigma] \to \sigma) \to \mu\alpha\rho \to \sigma$ instead of this rule (μ-E). But the present notation allows a uniform treatment of eliminations as expressed by the vector notation already introduced for every system in "the basic framework".

[2]$\wedge$ denotes the infimum of two lattice elements.

[3]Again it would be closer to standard presentations of e.g. Gödel's T if instead of the rule we had a recursor constant $R_{\mu\alpha\rho,\sigma}$ of type $(\rho[\alpha := \mu\alpha\rho \times \sigma] \to \sigma) \to \mu\alpha\rho \to \sigma$ in the system. Many thanks to Felix Joachimski for encouraging me to use the elimination notation throughout which came up in a discussion with him.

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

We come to the introduction rule which is the rule pertaining to $\mathsf{lfp}(\Phi)$ being a pre-fixed-point of Φ:

(lfp-I) If $x \in \Phi(\mathsf{lfp}(\Phi))$, then $x \in \mathsf{lfp}(\Phi)$.

Note that although we always assumed Φ to be monotone, the rules (lfp-E$^{(+)}$) would have trivially been valid for any Φ if we had set $\mathsf{lfp}(\Phi) := \bigwedge U$. ($\mathsf{lfp}$-I) does not accord with this setting. Hence, we pursue the following approach which is typical of logic: We assume $\mathsf{lfp}(\Phi)$ always to exist (in a formal system, we would introduce some predicate constant). We do not require the monotonicity of Φ! Then we describe $\mathsf{lfp}(\Phi)$ axiomatically by (lfp-E$^{(+)}$) and the following refined

(lfp-I-mon) If $x \in \Phi(\mathsf{lfp}(\Phi))$ and Φ monotone, then $x \in \mathsf{lfp}(\Phi)$.

Hence, for monotone Φ we may use Tarski's theorem to construct $\mathsf{lfp}(\Phi)$, and the trivial $\bigwedge U$ otherwise.
Φ monotone means $\forall M \forall M'.(\forall x.x \in M \Rightarrow x \in M') \Rightarrow \forall y(y \in \Phi(M) \Rightarrow y \in \Phi(M'))$. Hence, by our usual interpretation we can motivate the rule of μ-introduction for EMIT

(μ-I) If $t^{\rho[\alpha:=\mu\alpha\rho]} \in \Lambda$ and $m^{\forall\alpha\forall\beta.(\alpha\to\beta)\to\rho\to\rho[\alpha:=\beta]} \in \Lambda$ (with $\beta \notin \{\alpha\} \cup \mathsf{FV}(\rho)$), then $(C_{\mu\alpha\rho} mt)^{\mu\alpha\rho} \in \Lambda$.

The term m is the witness of monotonicity for $\lambda\alpha\rho$. The monotonicity witnesses are thus constructed simultaneously with the terms (they are terms) instead of being given from outside. The idea of the systems ESMIT is to free the introduction rule from the monotonicity witness but to ensure that for every $\mu\alpha\rho$ in the system (those which are allowed have to be specified) there is a monotonicity witness given beforehand. This witness also has to be a term of the system and clearly, we have to restrict the use of (μ-I) (without the monotonicity witness) in the construction of those witnesses. The exact condition will be called "stratification".
A variant $\mathsf{varEMIT}$ of EMIT is introduced which is motivated by the observation that only special cases of the monotonicity of Φ are needed for justifying the β-rules of EMIT.
The system IMIT is dual to the system EMIT in the sense that the monotonicity witnesses are tied to the elimination rules instead of the introduction rule. (The rule (lfp-I) is valid if we set $\mathsf{lfp}(\Phi) := \bigwedge \emptyset$ and the monotonicity requirement in the elimination rules guarantees that Tarski's theorem applies for the construction of $\mathsf{lfp}(\Phi)$ validating all the rules in case Φ is monotone.)
The systems ISMIT and $\mathsf{varIMIT}$ are derived from the same considerations as ESMIT and $\mathsf{varEMIT}$ are derived from EMIT. If one mimicks the construction of $\mathsf{varEMIT}$ directly for IMIT, then β-reduction does not normalize!

4.1 The elimination-based term rewrite system **EMIT** of monotone inductive types

EMIT is "elimination-based" because μ-elimination is primitive in the sense that no monotonicity witness is needed there. It is a system of monotone inductive types because there is no syntactic restriction on forming $\mu\alpha\rho$ but the requirement of a monotonicity witness in the μ-introduction rule. For the system IMIT defined in section 4.4 elimination and introduction change roles in this description. It seems to be better compliant with the philosophical semantics of Martin-Löf's type theory in [ML84] if one fixes the monotonicity witness to the introduction rule because monotonicity is a part of the justification for the existence of an object in the type $\mu\alpha\rho$—it is the justification that $\mu\alpha\rho$ is studied at all!

4.1.1 Definition

The (second) definition of typed terms and their free variables for system eF is extended by the following clauses:

(μ-I) If $t^{\rho[\alpha:=\mu\alpha\rho]} \in \Lambda$ and $m^{\forall\alpha\forall\beta.(\alpha\to\beta)\to\rho\to\rho[\alpha:=\beta]} \in \Lambda$ (with $\beta \notin \{\alpha\} \cup \mathsf{FV}(\rho)$), then $(C_{\mu\alpha\rho}mt)^{\mu\alpha\rho} \in \Lambda$ and $\mathsf{FV}(C_{\mu\alpha\rho}mt) := \mathsf{FV}(m) \cup \mathsf{FV}(t)$.

(μ-E) If $r^{\mu\alpha\rho} \in \Lambda$ and $s^{\rho[\alpha:=\sigma]\to\sigma} \in \Lambda$, then $(rs)^\sigma \in \Lambda$ (also written as $r\mathrm{E}_\mu\sigma s$ or shorter $r\mathrm{E}_\mu s$) and $\mathsf{FV}(rs) := \mathsf{FV}(r) \cup \mathsf{FV}(s)$.

(μ-E$^+$) If $r^{\mu\alpha\rho} \in \Lambda$ and $s^{\rho[\alpha:=\mu\alpha\rho\times\sigma]\to\sigma} \in \Lambda$, then $(rs)^\sigma \in \Lambda$ (also written as $r\mathrm{E}^+_\mu\sigma s$ or shorter $r\mathrm{E}^+_\mu s$) and $\mathsf{FV}(rs) := \mathsf{FV}(r) \cup \mathsf{FV}(s)$.

This definition is again compatible with the renaming convention for bound type variables.
We freely use all of the notational machinery previously introduced for F and eF. If $r : \mu\alpha\rho$, we say that r is of μ-type. (The use of "inductive type" is discouraged because it should be used more generally for any type of the systems which have iteration or primitive recursion.)
We extend the naming convention by assuming that m always denotes a term.
As was mentioned after giving the very first inductive definition of types (for system F) it is our understanding that the rules freely generate the structure. If $\alpha \in \mathsf{FV}(\rho)$ then $\rho[\alpha := \sigma] \neq \rho[\alpha := \mu\alpha\rho \times \sigma]$ and therefore the types of the terms s in the rules (μ-E) and (μ-E$^+$) are different. Otherwise rs can be generated by both rules. But according to our free generation assumption we have to distinguish them. Hence, the notation $r\mathrm{E}_\mu s$ vs. $r\mathrm{E}^+_\mu s$ is nearer to the "true" terms. However, in theoretic studies on the system EMIT it is quite unusual to focus specifically on cases where $\alpha \notin \mathsf{FV}(\rho)$ and therefore the appearance of $\mu\alpha\rho$ in the string which represents the type of $s^{\rho[\alpha:=\mu\alpha\rho\times\sigma]\to\sigma}$ clearly indicates that the use of rule (μ-E$^+$) is intended.
Extend the recursive definition of $\mathsf{FTV}(r)$ by the following clauses:

(μ-I) $\mathsf{FTV}(C_{\mu\alpha\rho}mt) := \mathsf{FV}(\mu\alpha\rho) \cup \mathsf{FTV}(m) \cup \mathsf{FTV}(t)$.

(μ-E$^{(+)}$) $\mathsf{FTV}(rs) := \mathsf{FTV}(r) \cup \mathsf{FTV}(s)$.

The definition of $r[\vec{x}^{\vec{\rho}} := \vec{s}]$ together with the proof of $r^\rho[\vec{x}^{\vec{\rho}} := \vec{s}] : \rho$ and $\mathsf{FV}(r[\vec{x}^{\vec{\rho}} := \vec{s}]) = (\mathsf{FV}(r) \setminus \vec{x}^{\vec{\rho}}) \cup \bigcup\{\mathsf{FV}(s_i) | x_i^{\rho_i} \in \mathsf{FV}(r) \wedge x_i^{\rho_i} \neq x_j^{\rho_j} \text{ for } j < i\}$ (i. e., the statement of Lemma 2.10) is extended by the following clauses:

(μ-I) $(C_{\mu\alpha\rho}mt)[\vec{x}^{\vec{\rho}} := \vec{s}] := C_{\mu\alpha\rho}m[\vec{x}^{\vec{\rho}} := \vec{s}]t[\vec{x}^{\vec{\rho}} := \vec{s}]$. Proof obvious.

(μ-E) $(rs)[\vec{x}^{\vec{\rho}} := \vec{s}] := r[\vec{x}^{\vec{\rho}} := \vec{s}]\mathrm{E}_\mu s[\vec{x}^{\vec{\rho}} := \vec{s}]$. Proof obvious.

(μ-E$^+$) $(rs)[\vec{x}^{\vec{\rho}} := \vec{s}] := r[\vec{x}^{\vec{\rho}} := \vec{s}]\mathrm{E}^+_\mu s[\vec{x}^{\vec{\rho}} := \vec{s}]$. Proof obvious.

The definition of $r[\vec{\alpha} := \vec{\sigma}]$ together with the proof of $r^\rho[\vec{\alpha} := \vec{\sigma}] : \rho[\vec{\alpha} := \vec{\sigma}]$ and $\mathsf{FV}(r[\vec{\alpha} := \vec{\sigma}]) = \{y^{\tau[\vec{\alpha}:=\vec{\sigma}]} | y^\tau \in \mathsf{FV}(r)\}$ is extended by the following clauses:

(μ-I) $(C_{\mu\alpha\rho}mt)[\vec{\alpha} := \vec{\sigma}] := C_{(\mu\alpha\rho)[\vec{\alpha}:=\vec{\sigma}]}m[\vec{\alpha} := \vec{\sigma}]t[\vec{\alpha} := \vec{\sigma}]$. Proof obvious.

(μ-E) $(rs)[\vec{\alpha} := \vec{\sigma}] := r[\vec{\alpha} := \vec{\sigma}]\mathrm{E}_\mu s[\vec{\alpha} := \vec{\sigma}]$. Proof obvious.

(μ-E$^+$) $(rs)[\vec{\alpha} := \vec{\sigma}] := r[\vec{\alpha} := \vec{\sigma}]\mathrm{E}^+_\mu s[\vec{\alpha} := \vec{\sigma}]$. Proof obvious.

The statements made in section 2.1.2 are valid in EMIT, too.
The general definition of immediate subterms of a term gives the following extension for system EMIT: $C_{\mu\alpha\rho}mt$ has exactly the immediate subterms m and t and $r\mathrm{E}_\mu s$ and $r\mathrm{E}^+_\mu s$ have exactly the immediate subterms r and s.
Extend the inductive definition of $r\vec{s}$ for system eF as follows:

(μ-E) If $r\vec{s} : \mu\alpha\rho$ and $s : \rho[\alpha := \sigma] \to \sigma$, then $r(\vec{s}, s) := (r\vec{s})s$.

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

The statement of Lemma 2.23 holds also true for EMIT.

Lemma 4.1 (Head form) Every term has either exactly one of the forms given in Lemma 2.44 or exactly one of the following forms:

(μ-I) $C_{\mu\alpha\rho}mt$

(μ-R$^{(+)}$) $(C_{\mu\alpha\rho}mt)s\vec{s}$, subsuming both eliminations for μ

Proof Induction on terms. □

Extend the inductive definition of the set NF for system eF by the following clause:

(μ-I) If $C_{\mu\alpha\rho}mt$ is a term and $m \in \mathsf{NF}$ and $t \in \mathsf{NF}$, then $C_{\mu\alpha\rho}mt \in \mathsf{NF}$.

The statements made in section 2.1.4 are all valid in EMIT.
Define the relation $\mapsto$ as for system eF, but add the following clauses:

EMIT

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

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

In both rules we require that $x^{\mu\alpha\rho} \notin \mathsf{FV}(s)$. The rule (β_μ) is the reduction rule associated with iteration and (β^+_μ) represents primitive recursion.
As for system eF the objects on the right side are automatically terms if the respective object on the left is a term, and the types are equal (which is not as trivial as before and is the main proof-theoretic justification for these rules as was mentioned before in the discussion on the Curry-Howard isomorphism).
The relation $\to_{whd}$ is defined as for system F and has the same properties as stated there.
Extend the inductive definition of the relation $\to$ for system eF by the following clauses (and prove that if $r \to r'$, then r and r' have the same type and $\mathsf{FV}(r') \subseteq \mathsf{FV}(r)$)[4]:

(μ-I) If $m \to m'$, then $C_{\mu\alpha\rho}mt \to C_{\mu\alpha\rho}m't$. If $t \to t'$, then $C_{\mu\alpha\rho}mt \to C_{\mu\alpha\rho}mt'$.

(μ-E) If $r \to r'$, then $r\mathrm{E}_\mu s \to r'\mathrm{E}_\mu s$. If $s \to s'$, then $r\mathrm{E}_\mu s \to r\mathrm{E}_\mu s'$.

(μ-E$^+$) If $r \to r'$, then $r\mathrm{E}^+_\mu s \to r'\mathrm{E}^+_\mu s$. If $s \to s'$, then $r\mathrm{E}^+_\mu s \to r\mathrm{E}^+_\mu s'$.

(The proofs were trivial.)
The extra $\beta_\to$-redex on the right side of the β^+_μ-rule is useful for applications. In order to represent a function which is defined by primitive recursion on an inductively defined set represented by $\mu\alpha\rho$ with values in a set represented by σ, we assume a closed term s of type $\rho[\alpha := \mu\alpha\rho \times \sigma] \to \sigma$ and set the representing term $F := \lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma s$ which is a closed term of type $\mu\alpha\rho \to \sigma$. We have

$$F(C_{\mu\alpha\rho}mt) \mapsto_{\beta_\to} C_{\mu\alpha\rho}mt\mathrm{E}^+_\mu\sigma s \mapsto_{\beta^+_\mu} s(m(\mu\alpha\rho)(\mu\alpha\rho \times \sigma)(\lambda x^{\mu\alpha\rho}.\langle x, Fx\rangle)t)$$

and therefore the left term $\to^+$-reduces to the right term which is valuable in practice. (Subscripts to $\mapsto$ indicate the rule which is used.)
Extend the inductive definition of WN for system eF by the following clauses:

[4] See also footnote 4 on p. 19 concerning well-formedness.

(μ-I) If $C_{\mu\alpha\rho}mt$ is a term and $m \in \mathsf{WN}$ and $t \in \mathsf{WN}$, then $C_{\mu\alpha\rho}mt \in \mathsf{WN}$.

(β_μ) If $s\Big(m(\mu\alpha\rho)\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma s)t\Big)\vec{s} \in \mathsf{WN}$, $x^{\mu\alpha\rho} \notin \mathsf{FV}(s)$ and $(C_{\mu\alpha\rho}mt)\mathrm{E}_\mu\sigma s\vec{s}$ is a term, then $(C_{\mu\alpha\rho}mt)\mathrm{E}_\mu\sigma s\vec{s} \in \mathsf{WN}$.

(β_μ^+) If $s\Big(m(\mu\alpha\rho)(\mu\alpha\rho \times \sigma)\Big(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma s)x\rangle\Big)t\Big)\vec{s} \in \mathsf{WN}$, $x^{\mu\alpha\rho} \notin \mathsf{FV}(s)$ and $(C_{\mu\alpha\rho}mt)\mathrm{E}_\mu^+\sigma s\vec{s}$ is a term, then $(C_{\mu\alpha\rho}mt)\mathrm{E}_\mu^+\sigma s\vec{s} \in \mathsf{WN}$.

Again definition by recursion on WN is admissible. Extend the recursive definition of the function Ω from WN to Λ and the simultaneous proof of $r \to^* \Omega(r) \in \mathsf{NF}$ for $r \in \mathsf{WN}$ by the following clauses:

(μ-I) $\Omega(C_{\mu\alpha\rho}mt) := C_{\mu\alpha\rho}\Omega(m)\Omega(t)$.

(β_μ) $\Omega((C_{\mu\alpha\rho}mt)\mathrm{E}_\mu\sigma s\vec{s}) := \Omega\Big(s\Big(m(\mu\alpha\rho)\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma s)t\Big)\vec{s}\Big)$.

(β_μ^+) $\Omega((C_{\mu\alpha\rho}mt)\mathrm{E}_\mu^+ s\vec{s}) := \Omega\Big(s\Big(m(\mu\alpha\rho)(\mu\alpha\rho \times \sigma)\Big(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma s)x\rangle\Big)t\Big)\vec{s}\Big)$.

The proofs are always obvious.
Extend the definition of SN for system eF by the following clauses:

(μ-I) If $C_{\mu\alpha\rho}mt$ is a term and $m \in \mathsf{SN}$ and $t \in \mathsf{SN}$, then $C_{\mu\alpha\rho}mt \in \mathsf{SN}$.

(β_μ) If $s\Big(m(\mu\alpha\rho)\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma s)t\Big)\vec{s} \in \mathsf{SN}$, $x^{\mu\alpha\rho} \notin \mathsf{FV}(s)$ and $(C_{\mu\alpha\rho}mt)\mathrm{E}_\mu\sigma s\vec{s}$ is a term, then $(C_{\mu\alpha\rho}mt)\mathrm{E}_\mu\sigma s\vec{s} \in \mathsf{SN}$.

(β_μ^+) If $s\Big(m(\mu\alpha\rho)(\mu\alpha\rho \times \sigma)\Big(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma s)x\rangle\Big)t\Big)\vec{s} \in \mathsf{SN}$, $x^{\mu\alpha\rho} \notin \mathsf{FV}(s)$ and $(C_{\mu\alpha\rho}mt)\mathrm{E}_\mu^+\sigma s\vec{s}$ is a term, then $(C_{\mu\alpha\rho}mt)\mathrm{E}_\mu^+\sigma s\vec{s} \in \mathsf{SN}$.

The only difference between the μ-clauses for SN and WN is the name of the defined set.
Every statement made in section 2.1.4 for system F (on nf, NF, wn, WN, Ω, SN and sn) is also true for EMIT.

4.1.2 Monotonicity without positivity

In this section we give evidence that monotonicity is more general than positivity. Note, however, that it is shown in this thesis that iteration on monotone inductive types may even be embedded into system F and that primitive recursion on monotone inductive types may be reduced (via impredicative encodings) to primitive recursion on positive inductive types.
The following example is due to Ulrich Berger (private communication, September 25, 1997).
For every type σ of system eF define $\neg\sigma := \sigma \to 0$. Define $\rho := (\neg\neg\alpha \to \alpha) \to \alpha$. Obviously, $\rho \in \mathsf{PosTypes}$, but $\alpha \notin \mathsf{Pos}(\rho)$ (where we view ρ as an inductive type formed without μ in order to have the notion of positivity).
Let u, v and w be term variables. Define the closed term r of system eF by

$$r := \Lambda\alpha\Lambda\beta\lambda x^{\alpha\to\beta}\lambda y^{\rho}\lambda z^{\neg\neg\beta\to\beta}.z(\lambda u^{\neg\beta}.u(x(y(\lambda v^{\neg\neg\alpha}.(v(\lambda w^{\alpha}.u(xw)))\mathrm{E}_0\alpha)))).$$

It is not hard to see that $r : \forall\alpha\forall\beta.(\alpha \to \beta) \to \rho \to \rho[\alpha := \beta]$.
This example should be contrasted with the following: Set $\rho' := (\alpha \to \alpha) \to \alpha$. Define the closed term r of system F by

$$r := \Lambda\alpha\Lambda\beta\lambda x^{\alpha\to\beta}\lambda y^{\rho'}\lambda z^{\beta\to\beta}.x(y(\lambda u^{\alpha}u)).$$

It is very easy to see that $r : \forall\alpha\forall\beta.(\alpha \to \beta) \to \rho' \to \rho'[\alpha := \beta]$. This second example is not convincing because there are closed terms $s : \rho' \to \alpha$ and $t : \alpha \to \rho'$ and trivially $\alpha \in \mathsf{Pos}(\alpha)$. (Set $s := \lambda y^{\rho'}.(y(\lambda u^{\alpha}u))$ and $t := \lambda x^{\alpha}\lambda y^{\alpha\to\alpha}x$.)
The first example does not have the tautology $\alpha \to \alpha$ in the premise of the implication but $\neg\neg\alpha \to \alpha$ which is only a tautology in classical logic (it defines classical logic).
There is no hope for a type $\tau \in \mathsf{PosTypes}$ with $\alpha \in \mathsf{Pos}(\tau)$ such that there are closed terms $s : \rho \to \tau$ and $t : \tau \to \rho$.
The example can be streamlined (Ulrich Berger, private communication, October 1, 1997) to only use the $\to$-fragment of minimal propositional logic, i. e., where only the type constructor $\to$ is used and not 0: ρ is replaced by $(((\alpha \to \sigma) \to \alpha) \to \alpha) \to \alpha$ for an arbitrary σ with $\alpha \notin \mathsf{FV}(\sigma)$. With the notation for the Peirce formula of Lemma 2.42 this type is $\mathsf{peirce}(\sigma) \to \alpha$. And by Corollary 2.43 we know that this example is essentially the same as the first one.
If we set $\tau := \mu\alpha.\mathsf{peirce}(\sigma) \to \alpha$, we get a type which is not inhabited: There is a closed term s of type $((((0 \to \sigma) \to 0) \to 0) \to 0) \to 0$, namely

$$s := \lambda z^{((((0\to\sigma)\to 0)\to 0)\to 0)}.z(\lambda y^{(0\to\sigma)\to 0}.(\lambda w^{0}.w\mathrm{E}_0\sigma)).$$

Therefore we have the closed term $\lambda x^{\tau}.x\mathrm{E}_\mu 0s : \tau \to 0$. Because 0 is not inhabited, neither is τ. But we may define $\mathsf{bizarre}(\sigma) := \mu\alpha.1 + (\mathsf{peirce}(\sigma) \to \alpha)$ and get an inhabited type[5] which is monotone and not positive.
Note that $\lambda\alpha.\alpha \to 1$ is trivially monotone and not positive. The type $\mu\alpha.\alpha \to 1$ will be dicussed in appendix B.

4.2 The term rewrite systems **ESMIT** of elimination-based selected monotone inductive types

These systems were invented to clarify the following observation: When trying to embed PIT (to be defined in section 5.1.1) into EMIT apparently the specific definition of $\mathsf{map}_{\lambda\alpha\rho}$ (and $\mathsf{comap}_{\lambda\alpha\rho}$) does not enter the argument. Due to interleaving those map-terms also have to use the term rules for μ-types. And it is intuitively clear that the monotonicity witness used for defining the reduction behaviour of a type should not make use of this very type. A deeper analysis shows that the properties needed are exactly stratification and uniformity introduced below.

4.2.1 Definition

Let MTypes (for "monotone types") be a subset of Types which is closed under substitution and the type-forming operations of system eF, i.e.

(V) If $\alpha \in \mathsf{Typevars}$, then $\alpha \in \mathsf{MTypes}$.

($\circ$) If $\rho \in \mathsf{MTypes}$ and $\sigma \in \mathsf{MTypes}$, then $\rho \circ \sigma \in \mathsf{MTypes}$ (for $\circ \in \{\to, \times, +\}$).

($\forall$) If $\alpha \in \mathsf{Typevars}$ and $\rho \in \mathsf{MTypes}$, then $\forall\alpha\rho \in \mathsf{MTypes}$.

(0) $0 \in \mathsf{MTypes}$.

(1) $1 \in \mathsf{MTypes}$.

[5] Note that this type need not be trivial. The type $\mathsf{nat} = \mu\alpha.1 + \alpha$ is much more interesting than $\mu\alpha\alpha$ which is also empty because $0 \to 0$ is inhabited.

(sub) If $\rho, \vec{\sigma} \in \mathsf{MTypes}$, then $\rho[\vec{\alpha} := \vec{\sigma}] \in \mathsf{MTypes}$.

Prominent examples for MTypes are the sets $\mathsf{SPosTypes}$ and $\mathsf{PosTypes}$ of section 3.2.
We set up the naming convention that in the systems ESMIT which are now being defined, the symbols ρ, σ and τ (also with decoration) denote elements of MTypes.
Define a term system by extending the restriction of eF which is derived from eF by taking the types ρ, σ and τ only out of MTypes as follows: Add the clauses

(μ-I) If $\mu\alpha\rho \in \mathsf{MTypes}$ and $t^{\rho[\alpha:=\mu\alpha\rho]} \in \Lambda$, then $(C_{\mu\alpha\rho}t)^{\mu\alpha\rho} \in \Lambda$ and $\mathsf{FV}(C_{\mu\alpha\rho}t) := \mathsf{FV}(t)$.

(μ-E) If $r^{\mu\alpha\rho} \in \Lambda$ and $s^{\rho[\alpha:=\sigma]\to\sigma} \in \Lambda$, then $(rs)^{\sigma} \in \Lambda$ (also written as $r\mathrm{E}_\mu\sigma s$ or shorter $r\mathrm{E}_\mu s$) and $\mathsf{FV}(rs) := \mathsf{FV}(r) \cup \mathsf{FV}(s)$.

(μ-E$^+$) If $r^{\mu\alpha\rho} \in \Lambda$ and $s^{\rho[\alpha:=\mu\alpha\rho\times\sigma]\to\sigma} \in \Lambda$ then $(rs)^{\sigma} \in \Lambda$ (also written as $r\mathrm{E}^+_\mu\sigma s$ or shorter $r\mathrm{E}^+_\mu s$) and $\mathsf{FV}(rs) := \mathsf{FV}(r) \cup \mathsf{FV}(s)$.

Lemma 4.2 If $\pi \in \mathsf{Types}$ and $r^\pi \in \Lambda$, then $\pi \in \mathsf{MTypes}$.

Proof Obvious. Use the closure properties imposed on MTypes (and the naming convention!).□

We use the notational machinery as for EMIT including the convention concerning the disambiguation of rs into $r\mathrm{E}_\mu s$ and $r\mathrm{E}^+_\mu s$.
Extend the recursive definition of $\mathsf{FTV}(r)$ by the following clauses:

(μ-I) $\mathsf{FTV}(C_{\mu\alpha\rho}t) := \mathsf{FV}(\mu\alpha\rho) \cup \mathsf{FTV}(t)$.

(μ-E) $\mathsf{FTV}(rs) := \mathsf{FTV}(r) \cup \mathsf{FTV}(s)$.

(μ-E$^+$) $\mathsf{FTV}(rs) := \mathsf{FTV}(r) \cup \mathsf{FTV}(s)$.

The definition of $r[\vec{x}^{\vec{\rho}} := \vec{s}]$ together with the proof of $r^\rho[\vec{x}^{\vec{\rho}} := \vec{s}] : \rho$ and $\mathsf{FV}(r[\vec{x}^{\vec{\rho}} := \vec{s}]) = (\mathsf{FV}(r) \setminus \vec{x}^{\vec{\rho}}) \cup \bigcup\{\mathsf{FV}(s_i) | x_i^{\rho_i} \in \mathsf{FV}(r) \wedge x_i^{\rho_i} \neq x_j^{\rho_j} \text{ for } j < i\}$ (i. e., the statement of Lemma 2.10) is extended by the following clauses:

(μ-I) $(C_{\mu\alpha\rho}t)[\vec{x}^{\vec{\rho}} := \vec{s}] := C_{\mu\alpha\rho}t[\vec{x}^{\vec{\rho}} := \vec{s}]$. Proof obvious.

(μ-E) $(rs)[\vec{x}^{\vec{\rho}} := \vec{s}] := r[\vec{x}^{\vec{\rho}} := \vec{s}]\mathrm{E}_\mu s[\vec{x}^{\vec{\rho}} := \vec{s}]$. Proof obvious.

(μ-E$^+$) $(rs)[\vec{x}^{\vec{\rho}} := \vec{s}] := r[\vec{x}^{\vec{\rho}} := \vec{s}]\mathrm{E}^+_\mu s[\vec{x}^{\vec{\rho}} := \vec{s}]$. Proof obvious.

The definition of $r[\vec{\alpha} := \vec{\sigma}]$ together with the proof of $r^\rho[\vec{\alpha} := \vec{\sigma}] : \rho[\vec{\alpha} := \vec{\sigma}]$ and $\mathsf{FV}(r[\vec{\alpha} := \vec{\sigma}]) = \{y^{\tau[\vec{\alpha}:=\vec{\sigma}]} | y^\tau \in \mathsf{FV}(r)\}$ is extended by the following clauses:

(μ-I) $(C_{\mu\alpha\rho}t)[\vec{\alpha} := \vec{\sigma}] := C_{(\mu\alpha\rho)[\vec{\alpha}:=\vec{\sigma}]}t[\vec{\alpha} := \vec{\sigma}]$. Proof obvious. Note that we need the (sub)-rule of the restriction on MTypes.

(μ-E) $(rs)[\vec{\alpha} := \vec{\sigma}] := r[\vec{\alpha} := \vec{\sigma}]\mathrm{E}_\mu s[\vec{\alpha} := \vec{\sigma}]$. Proof obvious.

(μ-E$^+$) $(rs)[\vec{\alpha} := \vec{\sigma}] := r[\vec{\alpha} := \vec{\sigma}]\mathrm{E}^+_\mu s[\vec{\alpha} := \vec{\sigma}]$. Proof obvious.

The statements made in section 2.1.2 are valid in the systems ESMIT, too.
The general definition of immediate subterms of a term gives the following extension for the systems ESMIT: $C_{\mu\alpha\rho}t$ has exactly the immediate subterm t and $r\mathrm{E}_\mu s$ and $r\mathrm{E}^+_\mu s$ have exactly the immediate subterms r and s.
Extend the inductive definition of $r\vec{s}$ for system eF as follows:

(μ-E) If $r\vec{s} : \mu\alpha\rho$ and $s : \rho[\alpha := \sigma] \to \sigma$, then $r(\vec{s}, s) := (r\vec{s})s$.

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

The statement of Lemma 2.23 holds also true for the systems ESMIT.

Lemma 4.3 (Head form) Every term has either exactly one of the forms given in Lemma 2.44 or exactly one of the following forms:

(μ-I) $C_{\mu\alpha\rho}t$

(μ-R) $(C_{\mu\alpha\rho}t)s\vec{s}$, subsuming both eliminations for μ

Proof Induction on terms. □

Extend the inductive definition of the set NF for system eF by the following clause:

(μ-I) If $C_{\mu\alpha\rho}t$ is a term and $t \in \mathsf{NF}$, then $C_{\mu\alpha\rho}t \in \mathsf{NF}$.

The statements made in section 2.1.4 are all valid in the systems ESMIT.
A term rewrite system of elimination-based selected monotone inductive types (generic name is ESMIT) is given by a set MTypes as above and a family $\{\mathsf{map}_{\lambda\alpha\rho} | \mu\alpha\rho \in \mathsf{MTypes}\}$ where for every $\mu\alpha\rho \in \mathsf{MTypes}$ the object $\mathsf{map}_{\lambda\alpha\rho}$ is a closed term of type $\forall\alpha\forall\beta.(\alpha \to \beta) \to \rho \to \rho[\alpha := \beta]$ (with $\beta \notin \{\alpha\} \cup \mathsf{FV}(\rho)$) under the following conditions:

(C) Compatibility: The indexing of the family is compatible with the renaming convention for bound type variables. (This is understood, but mentioned once at this place.)

(U) Uniformity: $\mathsf{map}_{\lambda\alpha\rho}[\vec{\alpha} := \vec{\sigma}] = \mathsf{map}_{(\lambda\alpha\rho)[\vec{\alpha}:=\vec{\sigma}]}$.

(S) Stratification: There is a well-founded relation $\succ$ on the types of the form $\mu\alpha\rho$ such that whenever $C_{\mu\alpha'\rho'}t$ is a subterm of $\mathsf{map}_{\lambda\alpha\rho}$, then $\mu\alpha\rho \succ \mu\alpha'\rho'$.

Well-foundedness of a binary relation $>$ on a set M shall for our purposes be defined to be the following property: If $A \subseteq M$ and $\forall x \in M.(\forall y \in M.x > y \to y \in A) \to x \in A$, then $A = M$. (In the language of term rewrite systems this is nothing but the assumption that $>$ is strongly normalizing.)
The existence of the family $\{\mathsf{map}_{\lambda\alpha\rho} | \mu\alpha\rho \in \mathsf{MTypes}\}$ ensures that the abstracted types $\lambda\alpha\rho$ with $\mu\alpha\rho \in \mathsf{MTypes}$ may be "turned into functors". In [Loa97] this is exemplified in the special case of a predicative variant of SPIT (see section 5.2).
Define the relation $\mapsto$ as for system eF (but with types restricted to MTypes and the restriction of the terms as above), but add the following clauses:

ESMIT

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

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

In both rules we require that $x^{\mu\alpha\rho} \notin \mathsf{FV}(s)$.
As for system EMIT the objects on the right side are automatically terms if the respective object on the left is a term, and the types are equal (which needs the same reasoning as for EMIT).
The relation $\to_{whd}$ is defined as for system F and has the same properties as stated there.
Extend the inductive definition of the relation $\to$ for system eF by the following clauses (and prove that if $r \to r'$, then r and r' have the same type and $\mathsf{FV}(r') \subseteq \mathsf{FV}(r)$):

(μ-I) If $t \to t'$, then $C_{\mu\alpha\rho}t \to C_{\mu\alpha\rho}t'$.

(μ-E) If $r \to r'$, then $r\mathrm{E}_\mu s \to r'\mathrm{E}_\mu s$. If $s \to s'$, then $r\mathrm{E}_\mu s \to r\mathrm{E}_\mu s'$.

(μ-E$^+$) If $r \to r'$, then $r\mathrm{E}^+_\mu s \to r'\mathrm{E}^+_\mu s$. If $s \to s'$, then $r\mathrm{E}^+_\mu s \to r\mathrm{E}^+_\mu s'$.

(The proofs were trivial.) The rules above are called the congruence rules for μ.
Extend the inductive definition of WN for system eF by the following clauses:

(μ-I) If $C_{\mu\alpha\rho}t$ is a term and $t \in \mathsf{WN}$, then $C_{\mu\alpha\rho}t \in \mathsf{WN}$.

(β_μ) If $s\Big(\mathsf{map}_{\lambda\alpha\rho}(\mu\alpha\rho)\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma s)t\Big)\vec{s} \in \mathsf{WN}$, $x^{\mu\alpha\rho} \notin \mathsf{FV}(s)$ and $(C_{\mu\alpha\rho}t)\mathrm{E}_\mu\sigma s\vec{s}$ is a term, then $(C_{\mu\alpha\rho}t)\mathrm{E}_\mu\sigma s\vec{s} \in \mathsf{WN}$.

(β^+_μ) If $s\Big(\mathsf{map}_{\lambda\alpha\rho}(\mu\alpha\rho)(\mu\alpha\rho\times\sigma)\Big(\lambda x^{\mu\alpha\rho}.\langle x,(\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma s)x\rangle\Big)t\Big)\vec{s} \in \mathsf{WN}$, $x^{\mu\alpha\rho} \notin \mathsf{FV}(s)$ and $(C_{\mu\alpha\rho}t)\mathrm{E}^+_\mu\sigma s\vec{s}$ is a term, then $(C_{\mu\alpha\rho}t)\mathrm{E}^+_\mu\sigma s\vec{s} \in \mathsf{WN}$.

Again definition by recursion on WN is admissible. Extend the recursive definition of the function Ω from WN to Λ and the simultaneous proof of $r \to^* \Omega(r) \in \mathsf{NF}$ for $r \in \mathsf{WN}$ by the following clauses:

(μ-I) $\Omega(C_{\mu\alpha\rho}t) := C_{\mu\alpha\rho}\Omega(t)$.

(β_μ) $\Omega((C_{\mu\alpha\rho}t)\mathrm{E}_\mu\sigma s\vec{s}) := \Omega\Big(s\Big(\mathsf{map}_{\lambda\alpha\rho}(\mu\alpha\rho)\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma s)t\Big)\vec{s}\Big)$.

(β^+_μ) $\Omega((C_{\mu\alpha\rho}t)\mathrm{E}^+_\mu s\vec{s}) := \Omega\Big(s\Big(\mathsf{map}_{\lambda\alpha\rho}(\mu\alpha\rho)(\mu\alpha\rho\times\sigma)\Big(\lambda x^{\mu\alpha\rho}.\langle x,(\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma s)x\rangle\Big)t\Big)\vec{s}\Big)$.

The proofs are always obvious.
Extend the definition of SN for system eF by the following clauses:

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

(β_μ) If $s\Big(\mathsf{map}_{\lambda\alpha\rho}(\mu\alpha\rho)\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma s)t\Big)\vec{s} \in \mathsf{SN}$, $x^{\mu\alpha\rho} \notin \mathsf{FV}(s)$ and $(C_{\mu\alpha\rho}t)\mathrm{E}_\mu\sigma s\vec{s}$ is a term, then $(C_{\mu\alpha\rho}t)\mathrm{E}_\mu\sigma s\vec{s} \in \mathsf{SN}$.

(β^+_μ) If $s\Big(\mathsf{map}_{\lambda\alpha\rho}(\mu\alpha\rho)(\mu\alpha\rho\times\sigma)\Big(\lambda x^{\mu\alpha\rho}.\langle x,(\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma s)x\rangle\Big)t\Big)\vec{s} \in \mathsf{SN}$, $x^{\mu\alpha\rho} \notin \mathsf{FV}(s)$ and $(C_{\mu\alpha\rho}t)\mathrm{E}^+_\mu\sigma s\vec{s}$ is a term, then $(C_{\mu\alpha\rho}t)\mathrm{E}^+_\mu\sigma s\vec{s} \in \mathsf{SN}$.

The only difference between the μ-clauses for SN and WN is the name of the defined set.
Every statement made in section 2.1.4 for system F (on nf, NF, wn, WN, Ω, SN and sn) is also true for the systems ESMIT.
Note that the conditions (U) and (S) were not used in this section. Nevertheless to my mind they should be part of the definition.

4.2.2 Embedding the systems ESMIT in EMIT

The embedding of the types will be the trivial one, i. e., $\rho' := \rho$ for every type $\rho \in \mathsf{MTypes}$. Let Λ' be the set of terms of EMIT.
Inductively define the set ST of stratified terms of the system ESMIT as a subset of Λ as follows:

(eF) All of the term generation rules of eF (where Λ is replaced by ST).

(μ-I) If $t^{\rho[\alpha:=\mu\alpha\rho]} \in \mathsf{ST}$ and $\mathsf{map}_{\lambda\alpha\rho} \in \mathsf{ST}$, then $C_{\mu\alpha\rho}t \in \mathsf{ST}$.

(μ-E) If $r^{\mu\alpha\rho} \in \mathsf{ST}$ and $s^{\rho[\alpha:=\sigma]\to\sigma} \in \mathsf{ST}$, then $rs \in \mathsf{ST}$.

(μ-E$^+$) If $r^{\mu\alpha\rho} \in \mathsf{ST}$ and $s^{\rho[\alpha:=\mu\alpha\rho\times\sigma]\to\sigma} \in \mathsf{ST}$ then $rs \in \mathsf{ST}$.

This definition is well-parsed in the sense which was explained for the definition of WN. Hence, we may define functions on ST by recursion on the definition.
Define the function $-' : \mathsf{ST} \to \Lambda'$ by recursion on ST and simultaneously prove that for $r : \rho$ we have $r' : \rho$ and $\mathsf{FTV}(r') = \mathsf{FTV}(r)$ and $\mathsf{FV}(r') = \mathsf{FV}(r)$ as follows:

(eF) The homomorphic term rules for eF. (Because of $\rho' = \rho$ the rule ($\to$-I) may be simplified to $(\lambda x^\rho r)' := \lambda x^\rho r'$ without the additional assumption.)

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

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

(μ-E$^+$) $(r\mathrm{E}^+_\mu s)' := r'\mathrm{E}^+_\mu s'$.

(The proofs are trivial.) The rules (μ-E) and (μ-E$^+$) are called the homomorphic μ-elimination rules. (μ-E) is called **the** homomorphic μ-elimination rule and (μ-E$^+$) is called the homomorphic μ^+-elimination rule.

Lemma 4.4 Every (not necessarily proper) subterm r of $\mathsf{map}_{\lambda\alpha\rho}$ is in ST.

Proof Main induction on $\mu\alpha\rho$ along $\succ$ coming from our assumption (S). Side induction on r (i. e., on the definition of Λ). All the term formation rules of eF and (μ-E) and (μ-E$^+$) are dealt with by the side induction hypothesis. E. g. ($\to$-E): Let $r\mathrm{E}_\to s$ be a subterm of $\mathsf{map}_{\lambda\alpha\rho}$. Then r and s are subterms of $\mathsf{map}_{\lambda\alpha\rho}$. By the side induction hypothesis, r and s are in ST. By the rule (eF) of the definition of ST, $rs \in \mathsf{ST}$. We come to the crucial case (μ-I): Let $C_{\mu\alpha'\rho'}t$ be a subterm of $\mathsf{map}_{\lambda\alpha\rho}$. By assumption (S) we have $\mu\alpha\rho \succ \mu\alpha'\rho'$. By the main induction hypothesis for $\mathsf{map}_{\lambda\alpha'\rho'}$ itself, we know that $\mathsf{map}_{\lambda\alpha'\rho'} \in \mathsf{ST}$. t is a subterm of $\mathsf{map}_{\lambda\alpha\rho}$. By the side induction hypothesis, $t \in \mathsf{ST}$. By the rule (μ-I) of the definition of ST, $C_{\mu\alpha'\rho'}t \in \mathsf{ST}$. □

Corollary 4.5 If $t \in \mathsf{ST}$ and $C_{\mu\alpha\rho}t \in \Lambda$, then $C_{\mu\alpha\rho}t \in \mathsf{ST}$. □

Corollary 4.6 $\mathsf{ST} = \Lambda$.

Proof ST has all the closure properties which define Λ. □

Therefore, $-'$ is defined on all terms. Obviously, it is injective on terms.
It should be mentioned that $\mathsf{ST} = \Lambda$ is not only a consequence of (S), but in fact equivalent to (S) (relative to the setting without (U) and (S)). The relation $\succ$ may be generated by assigning to a $\mu\alpha\rho$ the least ordinal θ (which exists under the assumption $\mathsf{ST} = \Lambda$) such that $C_{\mu\alpha\rho}x^{\rho[\alpha:=\mu\alpha\rho]}$ (with a fixed $x \in \mathsf{Vars}$) is in the application of the θ-iterate of the monotone operator associated with the inductive definition of ST to the empty set, and by comparing those ordinals.

Lemma 4.7 If $\mathsf{map}_{\lambda\alpha\rho} \in \mathsf{NF}$ for all $\mu\alpha\rho \in \mathsf{MTypes}$, then for every $r \in \mathsf{NF}$, also $r' \in \mathsf{NF}$.

Proof Induction on $r \in \mathsf{ST}$ using case analysis on $r \in \mathsf{NF}$. □

Lemma 4.8 $(r[\vec{x}^{\vec{\rho}} := \vec{s}^{\vec{\rho}}])' = r'[\vec{x}^{\vec{\rho}} := \vec{s}'']$ and $(r[\vec{\alpha} := \vec{\sigma}])' = r'[\vec{\alpha} := \vec{\sigma}]$.

Proof Induction on $r \in \mathsf{ST}$: All the cases except (μ-I) are straightforward. Case (μ-I):

$$((C_{\mu\alpha\rho}t)[\vec{x}^{\vec{\rho}} := \vec{s}])' = C_{\mu\alpha\rho}(\mathsf{map}_{\lambda\alpha\rho})'(t[\vec{x}^{\vec{\rho}} := \vec{s}])'.$$

$$(C_{\mu\alpha\rho}t)'[\vec{x}^{\vec{\rho}} := \vec{s}'] = C_{\mu\alpha\rho}(\mathsf{map}_{\lambda\alpha\rho})'[\vec{x}^{\vec{\rho}} := \vec{s}']t'[\vec{x}^{\vec{\rho}} := \vec{s}'].$$

By induction hypothesis $(t[\vec{x}^{\vec{\rho}} := \vec{s}])' = t'[\vec{x}^{\vec{\rho}} := \vec{s}']$. By assumption $\mathsf{map}_{\lambda\alpha\rho}$ is closed, hence $(\mathsf{map}_{\lambda\alpha\rho})'$ is closed as was proved simultaneously with the definition of $-'$. (This part of the lemma did not need the induction on ST. Induction on $r \in \Lambda$ would have been sufficient.)

$$((C_{\mu\alpha\rho}t)[\vec{\alpha} := \vec{\sigma}])' = C_{(\mu\alpha\rho)[\vec{\alpha}:=\vec{\sigma}]}(\mathsf{map}_{(\lambda\alpha\rho)[\vec{\alpha}:=\vec{\sigma}]})'(t[\vec{\alpha} := \vec{\sigma}])'.$$

$$(C_{\mu\alpha\rho}t)'[\vec{\alpha} := \vec{\sigma}] = C_{(\mu\alpha\rho)[\vec{\alpha}:=\vec{\sigma}]}(\mathsf{map}_{\lambda\alpha\rho})'[\vec{\alpha} := \vec{\sigma}]t'[\vec{\alpha} := \vec{\sigma}].$$

By induction hypothesis $(t[\vec{\alpha} := \vec{\sigma}])' = t'[\vec{\alpha} := \vec{\sigma}]$ and $(\mathsf{map}_{\lambda\alpha\rho}[\vec{\alpha} := \vec{\sigma}])' = (\mathsf{map}_{\lambda\alpha\rho})'[\vec{\alpha} := \vec{\sigma}]$. We need to show $(\mathsf{map}_{(\lambda\alpha\rho)[\vec{\alpha}:=\vec{\sigma}]})' = (\mathsf{map}_{\lambda\alpha\rho}[\vec{\alpha} := \vec{\sigma}])'$. Due to the injectivity of $-'$ this is equivalent to (U) which we assumed in the definition of ESMIT. □

Lemma 4.9 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 Lemma 4.8. The cases ($\beta_\times$) and (β_+) are obvious. The cases (β_μ) and (β_μ^+) are easy. Check e. g.

$$\begin{aligned} ((C_{\mu\alpha\rho}t)\mathrm{E}_\mu\sigma s)' &= (C_{\mu\alpha\rho}(\mathsf{map}_{\lambda\alpha\rho})'t')\mathrm{E}_\mu\sigma s' \\ &\mapsto s'\Big((\mathsf{map}_{\lambda\alpha\rho})'(\mu\alpha\rho)\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma s')t'\Big) \\ &= \Big(s\Big(\mathsf{map}_{\lambda\alpha\rho}(\mu\alpha\rho)\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma s)t\Big)\Big)' \end{aligned}$$

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

This implies that $-'$ is an embedding of ESMIT in EMIT.

4.2.3 The term rewrite systems ESMITit and ESMITrec

The systems ESMITit are derived from ESMIT just by forbidding the use of the term generation rule (μ-E$^+$) in the terms $\mathsf{map}_{\lambda\alpha\rho}$ (for $\mu\alpha\rho \in \mathsf{MTypes}$). Consequently the systems ESMITrec are defined by ruling out the term generation rule (μ-E) for the definition of the terms $\mathsf{map}_{\lambda\alpha\rho}$.

4.2.4 The term rewrite systems ESMIT+ex

The term rewrite systems ESMIT+ex are based on the types as described in section 3.4. Let MTypes be a subset of this set of types which is closed under substitution and the type-forming operations, i. e., MTypes is supposed to satisfy the conditions described in section 4.2.1 and

($\exists$) If $\alpha \in \mathsf{Typevars}$ and $\rho \in \mathsf{MTypes}$, then $\exists\alpha\rho \in \mathsf{MTypes}$.

The most prominent examples are the sets SPosTypes and PosTypes of section 3.4.
The term system is defined by reinterpreting the term formation rules for ESMIT in the extended type system and by adding the clauses ($\exists$-I) and ($\exists$-E) of section 2.4.2 (also interpreted in the extended type system).
A term rewrite system ESMIT+ex is given by MTypes and by a family $\{\mathsf{map}_{\lambda\alpha\rho} | \mu\alpha\rho \in \mathsf{MTypes}\}$ where again for every $\mu\alpha\rho \in \mathsf{MTypes}$ the object $\mathsf{map}_{\lambda\alpha\rho}$ is a closed term (in the term system just defined) of type $\forall\alpha\forall\beta.(\alpha \to \beta) \to \rho \to \rho[\alpha := \beta]$ (with $\beta \notin \{\alpha\} \cup \mathsf{FV}(\rho)$) such that the conditions (C), (U) and (S) of section 4.2.1 are met.
The relation $\mapsto$ is defined by adding the clauses for system eF+ex to the rules (β_μ) and (β_μ^+) of system ESMIT. All the statements made in section 4.2.1 are true for ESMIT+ex where the definitions are simply the sum of the clauses for eF+ex and ESMIT.

4.3 The variant varEMIT of EMIT

A slight generalization of EMIT is defined. It is not intended to cover more interesting examples of inductive types but serves as a challenge to prove strong normalization directly (see appendix A). Moreover, the embeddings of EMIT into other systems simply also work for varEMIT and therefore they are carried out in the more general system.
In section 4.3.3 it is shown that monotone inductive types with iteration only are embedded into IT (and hence also in F) via an analysis of Tarski's fixed-point theorem. This result generalizes and sharpens (because also reduction is preserved) earlier results on the impredicative encoding of data types (e. g. [BB85] where simultaneous non-interleaved strictly positive inductive types are studied, and I believe that simultaneous inductive types may easily be represented in EMIT via interleaving). Compare also with the presentation of impredicative encodings in [GLT89] and with [Lei90, p. 306] for the encoding (irrespective of reduction behaviour) in the case of positive inductive types.

4.3.1 Definition

The reduction rules of system EMIT use the term m which comes with the rule (μ-I) only with first type argument $\mu\alpha\rho$. In order to get maximal generality we define the variant varEMIT of EMIT by considering terms m of type

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

instead of the type

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

(note the renaming of variables which took place).
In the following only changes to EMIT are given. All the statements on EMIT hold as well in varEMIT.
Changed clauses in the definition of terms:

(μ-I) If $t^{\rho[\alpha:=\mu\alpha\rho]} \in \Lambda$ and $m^{\forall\alpha.(\mu\alpha\rho\to\alpha)\to\rho[\alpha:=\mu\alpha\rho]\to\rho} \in \Lambda$, then $(C_{\mu\alpha\rho}mt)^{\mu\alpha\rho} \in \Lambda$ and $\mathsf{FV}(C_{\mu\alpha\rho}mt) := \mathsf{FV}(m) \cup \mathsf{FV}(t)$.

Changed clauses in the definition of reduction:

varEMIT

$$\begin{array}{ll} (\beta_\mu) & (C_{\mu\alpha\rho}mt)\mathrm{E}_\mu\sigma s \mapsto s\Big(m\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma s)t\Big) \\ (\beta_\mu^+) & (C_{\mu\alpha\rho}mt)\mathrm{E}_\mu^+\sigma s \mapsto s\Big(m(\mu\alpha\rho \times \sigma)\Big(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma s)x\rangle\Big)t\Big) \end{array}$$

We again require $x^{\mu\alpha\rho} \notin \mathsf{FV}(s)$ in both rules. All the other definitions for system EMIT may be adapted very easily.

4.3.2 Embedding EMIT in varEMIT

An embedding $-'$ of EMIT in varEMIT is straightforward to define: Set $\rho' := \rho$ and for every term r^ρ of system EMIT define the term r' of system varEMIT by recursion on r and simultaneously prove that $r' : \rho$ and $\mathsf{FTV}(r') = \mathsf{FTV}(r)$ and $\mathsf{FV}(r') = \mathsf{FV}(r)$ as follows:

(eF) The homomorphic term rules for eF.

(μ-I) $(C_{\mu\alpha\rho}mt)' := C_{\mu\alpha\rho}(m'\mu\alpha\rho)t'$.

(μ-E) The homomorphic μ-elimination rules.

(The proofs are obvious.) The statements of Lemma 4.8 and Lemma 4.9 are true for this definition and are proved by induction on r resp. by induction on $r \to \hat{r}$. Only (β_μ) and (β_μ^+) deserve attention. Check e. g.

$$\begin{array}{lcl}(\beta_\mu^+):\quad ((C_{\mu\alpha\rho}mt)\mathrm{E}_\mu^+\sigma s)' & = & (C_{\mu\alpha\rho}(m'\mu\alpha\rho)t')\mathrm{E}_\mu^+\sigma s' \\ & \mapsto & s'\Big((m'\mu\alpha\rho)(\mu\alpha\rho\times\sigma)\Big(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma s')x\rangle\Big)t'\Big) \\ & = & \Big(s\Big(m(\mu\alpha\rho)(\mu\alpha\rho\times\sigma)\Big(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma s)x\rangle\Big)t\Big)\Big)'.\end{array}$$

4.3.3 Embedding varEMIT–rec in IT

Let us prove Tarski's fixed-point theorem (chapter 3). $M \in U$ is a post-fixed-point of Φ is defined to mean $M \le \Phi(M)$. Set

$$\mu_\Phi := \bigwedge\{M \in U | \Phi(M) \le M\}.$$

By definition, we have that $\mu_\Phi \le M$ for all pre-fixed-points M of Φ. Show that μ_Φ is itself a pre-fixed-point of Φ. Let M be a pre-fixed-point of Φ. Then $\mu_\Phi \le M$. Because Φ is monotone, this implies $\Phi(\mu_\Phi) \le \Phi(M)$. Because M is a pre-fixed-point of Φ and $\le$ is transitive, $\Phi(\mu_\Phi) \le M$ follows. Hence, $\Phi(\mu_\Phi) \le M$ for every pre-fixed-point M of Φ and therefore $\Phi(\mu_\Phi) \le \mu_\Phi$ by definition of μ_Φ. We conclude that μ_Φ is the least pre-fixed-point of Φ.
μ_Φ is also a post-fixed-point[6] of Φ (and consequently the least fixed point): We have to show that $\mu_\Phi \le \Phi(\mu_\Phi)$. By definition of μ_Φ, it suffices to show that $\Phi(\mu_\Phi)$ is a pre-fixed-point of Φ, i. e., $\Phi(\Phi(\mu_\Phi)) \le \Phi(\mu_\Phi)$. This follows from the first part of the proof and the monotonicity of Φ.
Let now $(U, \subseteq)$ be a complete lattice of sets. We rearrange the above proof in order to give the intuition for this section's embedding.
Firstly, by only applying ($\bigwedge$-E) to the definition of μ_Φ we get

(μ_Φ-E) If $x \in \mu_\Phi$ and $\Phi(M) \subseteq M$, then $x \in M$.

Secondly, we get

(μ_Φ-I) If Φ is monotone and $x \in \Phi(\mu_\Phi)$, then $x \in \mu_\Phi$.

via the following proof: Let Φ be monotone and $x \in \Phi(\mu_\Phi)$. We show $x \in \mu_\Phi$ using ($\bigwedge$-I). Let $M \in U$, $\Phi(M) \subseteq M$ and $y \in \Phi(\mu_\Phi)$. We show that $y \in M$. Let $z \in \mu_\Phi$. By ($\bigwedge$-E) and $\Phi(M) \subseteq M$, $z \in M$. Hence, $\mu_\Phi \subseteq M$. By monotonicity of Φ and $y \in \Phi(\mu_\Phi)$, $y \in \Phi(M)$. Due to $\Phi(M) \subseteq M$, $y \in M$. Hence we may apply ($\bigwedge$-I) using $x \in \Phi(\mu_\Phi)$ and get $x \in \mu_\Phi$.
Note that monotonicity is only used to conclude $\Phi(\mu_\Phi) \subseteq \Phi(M)$ from $\mu_\Phi \subseteq M$ and hence we may model the proof even in varEMIT instead of EMIT.
The definition of μ_Φ and the proofs of (μ_Φ-E) and (μ_Φ-I) are reflected in the following clauses of the embedding of varEMIT–rec in IT:

(μ) $(\mu\alpha\rho)' := i\alpha.\rho' \to \alpha$.

(μ-E) $(r^{\mu\alpha\rho}s^{\rho[\alpha:=\sigma]\to\sigma})' := r'\mathrm{E}_i\sigma's'$.

[6] Unfortunately, this fact cannot be used to embed fixed-point types into IT because of bad reduction behaviour. See the discussion in appendix B.

(μ-I) $(C_{\mu\alpha\rho}m^{\forall\alpha.(\mu\alpha\rho\to\alpha)\to\rho[\alpha:=\mu\alpha\rho]\to\rho}t^{\rho[\alpha:=\mu\alpha\rho]})' := C_{i\alpha.\rho'\to\alpha}\ell t'$ setting

$$\ell := \Lambda\alpha\lambda z^{\rho'\to\alpha}\lambda y^{\rho'[\alpha:=(\mu\alpha\rho)']}.z\Big(m'\alpha(\lambda x^{(\mu\alpha\rho)'}.x\mathrm{E}_i\alpha z)y\Big),$$

where we assume that $\alpha \notin \mathsf{FTV}(m)$ and require $z, y \notin \mathsf{FV}(m)$.

More formally: Let varEMIT–rec (varEMIT minus recursion) be the system which is derived from varEMIT by leaving out the term generation rule (μ-E^+) and the reduction rule (β_μ^+).
For every type ρ of system varEMIT–rec define the type ρ' of system IT by recursion on ρ and simultaneously prove that $\mathsf{FV}(\rho') = \mathsf{FV}(\rho)$ as follows:

(eF) The homomorphic type rules for eF.

(μ) $(\mu\alpha\rho)' := i\alpha.\rho' \to \alpha$.

(The proofs were obvious.) This definition is compatible with the variable conventions for the systems varEMIT–rec and IT.

Lemma 4.10 $(\rho[\vec{\alpha} := \vec{\sigma}])' = \rho'[\vec{\alpha} := \vec{\sigma}']$.

Proof Induction on ρ. □

For every term r^ρ of system varEMIT–rec define the term r' of system IT by recursion on r and simultaneously prove that $r' : \rho'$ and $\mathsf{FTV}(r') = \mathsf{FTV}(r)$ and $\mathsf{FV}(r') = \{x^{\sigma'} | x^\sigma \in \mathsf{FV}(r)\}$ as follows:

(eF) The homomorphic term rules for eF.

(μ-I) $(C_{\mu\alpha\rho}m^{\forall\alpha.(\mu\alpha\rho\to\alpha)\to\rho[\alpha:=\mu\alpha\rho]\to\rho}t^{\rho[\alpha:=\mu\alpha\rho]})' := C_{i\alpha.\rho'\to\alpha}\ell t'$ setting

$$\ell := \Lambda\alpha\lambda z^{\rho'\to\alpha}\lambda y^{\rho'[\alpha:=(\mu\alpha\rho)']}.z\Big(m'\alpha(\lambda x^{(\mu\alpha\rho)'}.x\mathrm{E}_i\alpha z)y\Big),$$

where we assume that $\alpha \notin \mathsf{FTV}(m)$ and require $z, y \notin \mathsf{FV}(m)$.

(μ-E) $(r^{\mu\alpha\rho}s^{\rho[\alpha:=\sigma]\to\sigma})' := r'\mathrm{E}_i\sigma's'$.

(The proofs were always obvious.) This definition is again compatible with the variable conventions for varEMIT–rec and IT.

Lemma 4.11 $(r[\vec{x}^{\vec{\rho}} := \vec{s}^{\vec{\rho}}])' = r'[\vec{x}^{\vec{\rho}'} := \vec{s}']$ and $(r[\vec{\alpha} := \vec{\sigma}])' = r'[\vec{\alpha} := \vec{\sigma}']$.

Proof Induction on r. □

Lemma 4.12 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 Lemma 4.11, $(\beta_\times)$ and (β_+) are trivial.

$$\begin{array}{lcl}(\beta_\mu):\quad ((C_{\mu\alpha\rho}mt)\mathrm{E}_\mu\sigma s)' & = & (C_{i\alpha.\rho'\to\alpha}\ell t')\mathrm{E}_i\sigma's' \quad \text{with } \ell \text{ as defined above} \\ & \mapsto & \ell\mathrm{E}_\forall\sigma's't' \\ & \to & (\lambda z^{\rho'[\alpha:=\sigma']\to\sigma'}\lambda y^{\rho'[\alpha:=(\mu\alpha\rho)']}.z(m'\sigma'(\lambda x^{(\mu\alpha\rho)'}.x\mathrm{E}_i\sigma'z)y))s't' \\ & \to & (\lambda y^{\rho'[\alpha:=(\mu\alpha\rho)']}.s'(m'\sigma'(\lambda x^{(\mu\alpha\rho)'}.x\mathrm{E}_i\sigma's')y))t' \\ & \to & s'(m'\sigma'(\lambda x^{(\mu\alpha\rho)'}.x\mathrm{E}_i\sigma's')t') \\ & = & \Big(s(m\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma s)t)\Big)'\end{array}$$

We used Lemma 2.16 and Corollary 2.2 for the second reduction step and Lemma 2.11 for the third and fourth. The other clauses are easily proved by using the induction hypothesis. □

4.4 The introduction-based term rewrite system **IMIT** of monotone inductive types

IMIT is "introduction-based" because the introduction rule is free from the monotonicity assumption.
The (second) definition of typed terms and their free variables for system eF is extended by the following clauses:

(μ-I) If $t^{\rho[\alpha:=\mu\alpha\rho]} \in \Lambda$, then $(C_{\mu\alpha\rho}t)^{\mu\alpha\rho} \in \Lambda$ and $\mathsf{FV}(C_{\mu\alpha\rho}t) := \mathsf{FV}(t)$.

(μ-E) If $r^{\mu\alpha\rho} \in \Lambda$, $m^{\forall\alpha\forall\beta.(\alpha\to\beta)\to\rho\to\rho[\alpha:=\beta]} \in \Lambda$ and $s^{\rho[\alpha:=\sigma]\to\sigma} \in \Lambda$, then $(rms)^\sigma \in \Lambda$ (also written as $r\mathrm{E}_\mu\sigma ms$ or shorter $r\mathrm{E}_\mu ms$) and $\mathsf{FV}(rs) := \mathsf{FV}(r) \cup \mathsf{FV}(m) \cup \mathsf{FV}(s)$.

(μ-E$^+$) If $r^{\mu\alpha\rho} \in \Lambda$, $m^{\forall\alpha\forall\beta.(\alpha\to\beta)\to\rho\to\rho[\alpha:=\beta]} \in \Lambda$ and $s^{\rho[\alpha:=\mu\alpha\rho\times\sigma]\to\sigma} \in \Lambda$ then $(rms)^\sigma \in \Lambda$ (also written as $r\mathrm{E}^+_\mu\sigma ms$ or shorter $r\mathrm{E}^+_\mu ms$) and $\mathsf{FV}(rs) := \mathsf{FV}(r) \cup \mathsf{FV}(m) \cup \mathsf{FV}(s)$.

In both elimination rules we require $\beta \notin \{\alpha\} \cup \mathsf{FV}(\rho)$. Therefore, this definition is again compatible with the renaming convention for bound type variables.
The only difference between the systems EMIT and IMIT is the place where the monotonicity witnesses are put in the terms. In EMIT monotonicity is needed in order to construct inhabitants of $\mu\alpha\rho$, in IMIT monotonicity is required for "using" inhabitants of $\mu\alpha\rho$, i. e., for defining functions on $\mu\alpha\rho$ by iteration or primitive recursion.
All the other definitions in section 4.1.1 may be simply derived from those for system EMIT by moving the term m from the position after the $C_{\mu\alpha\rho}$ to the position after E^+_μ and E_μ.
Define e. g. the relation $\mapsto$ by adding the following clauses to those for system eF:

IMIT

$$
\begin{array}{ll}
(\beta_\mu) & (C_{\mu\alpha\rho}t)\mathrm{E}_\mu\sigma ms \mapsto s\Big(m(\mu\alpha\rho)\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma ms)t\Big) \\
(\beta^+_\mu) & (C_{\mu\alpha\rho}t)\mathrm{E}^+_\mu\sigma ms \mapsto s\Big(m(\mu\alpha\rho)(\mu\alpha\rho\times\sigma)\Big(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma ms)x\rangle\Big)t\Big)
\end{array}
$$

(In both rules we now require that $x^{\mu\alpha\rho} \notin \mathsf{FV}(m) \cup \mathsf{FV}(s)$.)
Every statement made in section 2.1.4 for system F (on nf, NF, wn, WN, Ω, SN and sn) is also true for IMIT.

4.5 Relating iteration and primitive recursion

In the introduction to this chapter the rule (lfp-E$^+$) was derived from (lfp-E). Therefore we might expect that also primitive recursion may be expressed by iteration. This is not the case.
It is easy to derive (lfp-E) from (lfp-E$^+$). However, also iteration may not be satisfactorily expressed by primitive recursion (although there are very complicated reduction-preserving embeddings into the fragments without iteration).

4.5.1 Coding IMIT in IMIT–rec

This section shows why (μ-E$^+$) is not superfluous in IMIT (and also not in EMIT).
Let again $(U, \subseteq)$ be a complete lattice of sets. We reconsider how (lfp-E$^+$) is derived from (lfp-E) for monotone Φ:
Set $M' := \mathsf{lfp}(\Phi) \wedge M$. Assume $x \in \mathsf{lfp}(\Phi)$ and $\Phi(M') \subseteq M$. We first show $x \in M'$ by induction on $\mathsf{lfp}(\Phi)$. We only have to show $\Phi(M') \subseteq M'$. Let $y \in \Phi(M')$. Show that (1) $y \in \mathsf{lfp}(\Phi)$ and

(2) $y \in M$. Ad (1): Because Φ is monotone, $M' \subseteq \mathsf{lfp}(\Phi)$ and $y \in \Phi(M')$, we get $y \in \Phi(\mathsf{lfp}(\Phi))$. By (lfp-I) we get (1). Ad (2): Because $y \in \Phi(M')$ and $\Phi(M') \subseteq M$, we have $y \in M$ which is (2). Therefore $x \in M'$. Because $M' = \mathsf{lfp}(\Phi) \wedge M$, also $x \in M$.
Due to the use of Φ's monotonicity in the inductive step, this is clearly best expressed in the system IMIT which has monotonicity witnesses attached to (μ-E) and (μ-E$^+$). Our informal realizability method directly motivates the following clause of an encoding of IMIT in itself.

(μ-E$^+$) $(r^{\mu\alpha\rho}\mathrm{E}^+_\mu\sigma ms)' := (r'\mathrm{E}_\mu(\mu\alpha\rho\times\sigma)m'\hat{s})\mathsf{R}$ setting

$$\hat{s} := \lambda x^{\rho[\alpha:=\mu\alpha\rho\times\sigma]}.\langle C_{\mu\alpha\rho}\Big(m'(\mu\alpha\rho\times\sigma)(\mu\alpha\rho)(\lambda z^{\mu\alpha\rho\times\sigma}.z\mathsf{L})x\Big), s'x\rangle,$$

where we require that $x \notin \mathsf{FV}(m) \cup \mathsf{FV}(s)$.

Unfortunately, this encoding is not well-behaved with respect to reduction.
More formally: Let IMIT–rec (IMIT minus recursion) be the system which is derived from IMIT by leaving out the term generation rule (μ-E$^+$) and consequently the reduction rule (β^+_μ). Define for every term r^ρ of system IMIT the term r' of system IMIT–rec by recursion on r and simultaneously prove that $r' : \rho$ and $\mathsf{FTV}(r') = \mathsf{FTV}(r)$ and $\mathsf{FV}(r') = \mathsf{FV}(r)$ as follows:

(eF) The homomorphic term rules for eF.

(μ-I) The homomorphic μ-introduction rule.

(μ-E) The homomorphic μ-elimination rule.

(μ-E$^+$) $(r^{\mu\alpha\rho}\mathrm{E}^+_\mu\sigma ms)' := (r'\mathrm{E}_\mu(\mu\alpha\rho\times\sigma)m'\hat{s})\mathsf{R}$ setting

$$\hat{s} := \lambda x^{\rho[\alpha:=\mu\alpha\rho\times\sigma]}.\langle C_{\mu\alpha\rho}\Big(m'(\mu\alpha\rho\times\sigma)(\mu\alpha\rho)(\lambda z^{\mu\alpha\rho\times\sigma}.z\mathsf{L})x\Big), s'x\rangle,$$

where we require that $x \notin \mathsf{FV}(m) \cup \mathsf{FV}(s)$.

(The proofs are obvious.)
The statement of Lemma 4.8 (compatibility with substitution) also holds for $-'$, but we do not get an embedding:

$$
\begin{array}{rcl}
(C_{\mu\alpha\rho}t\mathrm{E}^+_\mu ms)' & = & (C_{\mu\alpha\rho}t'\mathrm{E}_\mu m'\hat{s})\mathsf{R} \quad \text{with } \hat{s} \text{ as above} \\
& \mapsto & \hat{s}\Big(m'(\mu\alpha\rho)(\mu\alpha\rho\times\sigma)(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu m'\hat{s})t'\Big)\mathsf{R} \\
& \to^2 & s'\Big(m'(\mu\alpha\rho)(\mu\alpha\rho\times\sigma)(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu m'\hat{s})t'\Big) \\
& \not\to^* & s'\Big(m'(\mu\alpha\rho)(\mu\alpha\rho\times\sigma)\Big(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma m's')x\rangle\Big)t'\Big)
\end{array}
$$

Instead of $\not\to^*$ it is in general (except trivial cases) also correct to write "not being in the symmetric closure $=_\beta$ of $\to^*$". There are two reasons: There is no induction hypothesis available which would give $\lambda x^{\mu\alpha\rho}.(x\mathrm{E}_\mu m'\hat{s})\mathsf{R} =_\beta \lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma m's'$. Moreover, we would in addition have to prove $(C_{\mu\alpha\rho}t'\mathrm{E}_\mu m'\hat{s})\mathsf{L} =_\beta C_{\mu\alpha\rho}t'$ which in turn would require functoriality properties of the term m' and also an induction hypothesis of the above kind. The first problem can be solved by introducing extensional equality instead of $=_\beta$, the second is then solvable only for positive inductive types (see the remarks in appendix B). However, there is no hope for finding term rewrite rules which make $-'$ an embedding.
For system EMIT there is even no hope for finding an encoding of (μ-E$^+$) via (μ-E) following the derivation of (lfp-E$^+$) from (lfp-E). Where should the monotonicity witness come from? In the lattice-theoretic situation this would require a rule

$$\forall x. x \in \mathsf{lfp}(\Phi) \Rightarrow \Phi \text{ monotone},$$

which cannot be a consequence of (lfp-I-mon), (lfp-E) and (lfp-E$^+$), because we can validate these rules interpreting $\mathsf{lfp}(\Phi)$ by Tarski's construction for monotone Φ and by $\bigwedge U$ otherwise. And it is of course possible to have complete lattices U of sets with $\bigwedge U \neq \emptyset$ also having non-monotone operators. These lattices refute the above rule. (Of course, in this situation we could give a trivial proof that $x \in \mathsf{lfp}(\Phi)$ and $\Phi(M') \subseteq M$ entail $x \in M$, because $\mathsf{lfp}(\Phi) \subseteq M$ for every $M \in U$. But such a non-uniform argument can never be expressed by means of EMIT.)

4.5.2 Coding IMIT in IMIT–it

This section shows why (μ-E) is not superfluous in IMIT(and also not in EMIT). But in principle it could be eliminated.
Let $(U, \leq)$ be a complete lattice. We derive (lfp-E) from (lfp-E$^+$) for monotone Φ:
Let $\Phi(M) \leq M$. We have to show that $\Phi(\mathsf{lfp}(\Phi) \wedge M) \leq M$. But trivially, $\mathsf{lfp}(\Phi) \wedge M \leq M$ and by monotonicity of Φ, $\Phi(\mathsf{lfp}(\Phi) \wedge M) \leq \Phi(M)$. We are done by applying transitivity of $\leq$ to the last formula and $\Phi(M) \leq M$.
Now let $(U, \subseteq)$ be a complete lattice of sets. We reformulate the above proof to allow reading off an encoding of (μ-E) by means of (μ-E$^+$):
Let $x \in \mathsf{lfp}(\Phi)$ and $\Phi(M) \subseteq M$. We show that $x \in M$ by extended induction on $\mathsf{lfp}(\Phi)$: Let $y \in \Phi(\mathsf{lfp}(\Phi) \wedge M)$. We have to show $y \in M$. Because Φ is monotone, $\mathsf{lfp}(\Phi) \wedge M \subseteq M$ and by the assumption on y, we get $y \in \Phi(M)$. Using $\Phi(M) \subseteq M$, we conclude $y \in M$.
Again the monotonicity of Φ is used in the induction step and therefore we first try to use this proof to get an encoding for IMIT. The following clause is read off the proof immediately:

(μ-E) $(r^{\mu\alpha\rho}\mathrm{E}_\mu\sigma ms)' := r'\mathrm{E}^+_\mu\sigma\Big(\lambda x^{\rho[\alpha:=\mu\alpha\rho\times\sigma]}.s'(m'(\mu\alpha\rho\times\sigma)\sigma(\lambda z^{\mu\alpha\rho\times\sigma}.z\mathsf{R})x)\Big)$, where we require that $x \notin \mathsf{FV}(m) \cup \mathsf{FV}(s)$.

Unfortunately, also this encoding is not well-behaved with respect to reduction.
More formally: Let IMIT–it (IMIT minus iteration) be the system which is derived from IMIT by leaving out the term generation rule (μ-E) and consequently the reduction rule (β_μ). Define for every term r^ρ of system IMIT the term r' of system IMIT–it by recursion on r and simultaneously prove that $r' : \rho$ and $\mathsf{FTV}(r') = \mathsf{FTV}(r)$ and $\mathsf{FV}(r') = \mathsf{FV}(r)$ as follows:

(eF) The homomorphic term rules for eF.

(μ-I) The homomorphic μ-introduction rule.

(μ-E) $(r^{\mu\alpha\rho}\mathrm{E}_\mu\sigma ms)' := r'\mathrm{E}^+_\mu\sigma\Big(\lambda x^{\rho[\alpha:=\mu\alpha\rho\times\sigma]}.s'(m'(\mu\alpha\rho\times\sigma)\sigma(\lambda z^{\mu\alpha\rho\times\sigma}.z\mathsf{R})x)\Big)$, where we require that $x \notin \mathsf{FV}(m) \cup \mathsf{FV}(s)$.

(μ-E$^+$) The homomorphic μ^+-elimination rule.

(The proofs are obvious.)
The statement of Lemma 4.8 (compatibility with substitution) also holds for $-'$, but we do not get an embedding (set $\hat{s} := \lambda x^{\rho[\alpha:=\mu\alpha\rho\times\sigma]}.s'(m'(\mu\alpha\rho\times\sigma)\sigma(\lambda z^{\mu\alpha\rho\times\sigma}.z\mathsf{R})x)$):
$(C_{\mu\alpha\rho}t\mathrm{E}_\mu ms)' = (C_{\mu\alpha\rho}t'\mathrm{E}^+_\mu m'\hat{s}) \mapsto$
$\hat{s}\Big(m'(\mu\alpha\rho)(\mu\alpha\rho\times\sigma)(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu m'\hat{s})x\rangle)t'\Big) \to$
$s'\Big(m'(\mu\alpha\rho\times\sigma)\sigma(\lambda z^{\mu\alpha\rho\times\sigma}.z\mathsf{R})\Big(m'(\mu\alpha\rho)(\mu\alpha\rho\times\sigma)(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu m'\hat{s})x\rangle)t'\Big)\Big)$.
The composition of $\lambda z^{\mu\alpha\rho\times\sigma}.z\mathsf{R}$ with $\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu m'\hat{s})x\rangle$ is $\beta_\to$-equal to $\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu m'\hat{s}$.
If m' were "functorial" one could in some sense get to

$$s'\Big(m'(\mu\alpha\rho)\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu m'\hat{s})t'\Big).$$

At first sight it may seem that exactly the same problems arise as with embedding IMIT in IMIT–rec. But here the situation is easier: One does not need some formal induction principle but only faces the functoriality problem. This is not solvable in general (for monotone inductive types). If one introduces extensional equality (exhibiting that formal induction principle) it is possible to show functoriality for the canonical monotonicity witnesses of PIT (in the equality theory). For NIPIT without sum types this is even possible with respect to the rewrite relation after adding some η-rules which do no harm to confluence and strong normalization. (For full NIPIT also permutative conversions have to be studied.) See also appendix B.
To sum up, we do not have an embedding and no hopes of finding an extension of the rewrite relation which settles the problem for IMIT.
We also cannot expect to even get an encoding of iteration via recursion in EMIT.
Having said all this it may seem paradoxical that **there are** reduction-preserving embeddings of IMIT in IMIT–it and of EMIT in EMIT–it (which is defined in the expected way). But these embeddings are very different from the above encoding. And they also affect the rule (μ-E$^+$) which will have no relation to the homomorphic μ^+-elimination rule.
For the argument we freely use facts spread over this thesis:

- IMIT embeds into varIMIT and varIMIT into MeIT and MeIT into MeIT–it.
- MeIT embeds into UVIT and UVIT into MePIT which is a subsystem of NISPIT+ex. NISPIT+ex is one of the systems ISMIT+ex (ISMIT extended by $\exists$, NISPIT+ex is defined as one of the systems ESMIT+ex but clearly is also an instance of ISMIT+ex).
- ISMIT embeds into IMIT.
- Therefore by extension of the proof that ISMIT embeds into IMIT also ISMIT+ex embeds into IMIT+ex (which is defined in the expected way).
- Clearly, the proof that eF+ex embeds into eF carries over to a proof that IMIT+ex embeds into IMIT.
- By the last four items, MeIT embeds into IMIT. By inspection of the proofs we note that the rule (μ-E) is always only used to deal with (μ-E) of some other system. Note that NISPIT+ex is a subsystem of PIT+ex not making use of (μ-E) in the definition of the canonical monotonicity witnesses because it is a system of non-interleaving μ-types! Therefore, also MeIT–it embeds into IMIT–it.
- Putting together the first item and the preceding one, we conclude that IMIT embeds into IMIT–it.

We see that the elimination of iteration happens in MeIT and that an awful lot of encoding[7] has to happen to go from IMIT to itself via MeIT. Note that the embedding of eF+ex in eF does not carry over to systems of selected monotone inductive types because the encoding of the existential quantifier is in general incompatible with the selected monotonicity witnesses.
The case of the embedding of EMIT in EMIT–it is somewhat easier because no existential quantifiers come into play:

- EMIT embeds into varEMIT and varEMIT into coMeIT and coMeIT into coMeIT–it.

[7] Of course, one could slightly optimize the embedding coming from the composition of all those embeddings by some normalization. Because $(\mu\alpha\rho)' = \mu\beta\forall\gamma.(\forall\alpha.(\alpha\to\beta)\times\rho'\to\gamma)\to\gamma$ this will still be quite complicate an embedding.

- coMeIT embeds into coMePIT[8] which is a subsystem of NIPIT. NIPIT is one of the systems ESMIT and every ESMIT embeds into EMIT.
- By inspection of the proofs we again see that the rule (μ-E) is always only used to deal with (μ-E) of some other system and that although NIPIT is a subsystem of PITit, (μ-E) is not used in the definition of the canonical monotonicity witnesses because it is a system of non-interleaving μ-types. Therefore, also coMeIT–it embeds into EMIT–it.
- We conclude that EMIT embeds into EMIT–it.

This time the elimination of iteration happens in coMeIT.
Hopefully these lines of argumentation motivate the reader to go through all of the embeddings presented in this thesis. They do serve a purpose. E. g. we may now conclude that iteration may also be eliminated in favour of recursion in all the concrete systems appearing in the above proofs because they show a circle of embeddings.
A similar solution to the problem of elimination of recursion by means of iteration is not in sight. Using the embedding of varEMIT–rec in IT (section 4.3.3) and some other embeddings we would arrive at embeddings of all the systems into system F which I strongly believe is impossible although I have not yet seen a proof.

4.6 The term rewrite systems ISMIT

These systems are dual to the systems ESMIT. This time one has to restrict the use of $\mu^{(+)}$-eliminations in the monotonicity witnesses.

4.6.1 Definition

The term rewrite systems ISMIT are defined to be the same as the systems ESMIT except that condition (S) of ESMIT is replaced by the following condition (S) of ISMIT:

(S) Stratification: There is a well-founded relation $\succ$ on the types of the form $\mu\alpha\rho$ such that whenever $r^{\mu\alpha'\rho'}s$ is a subterm of $\mathsf{map}_{\lambda\alpha\rho}$, then $\mu\alpha\rho \succ \mu\alpha'\rho'$.

As was remarked at the end of section 4.2.1, condition (S) is not used in that section and therefore all of the results therein are true for the systems ISMIT.

4.6.2 Embedding ISMIT in IMIT

The following embedding of the systems ISMIT in IMIT is derived by obvious modifications of the embedding of ESMIT in EMIT.
The embedding of the types is the trivial one, i. e., $\rho' := \rho$ for every type $\rho \in \mathsf{MTypes}$. Let Λ' be the set of terms of IMIT.
Inductively define the set ST of stratified terms of the system ISMIT as a subset of Λ as follows:

(eF) All of the term generation rules of eF (where Λ is replaced by ST).

(μ-I) If $t^{\rho[\alpha:=\mu\alpha\rho]} \in \mathsf{ST}$, then $C_{\mu\alpha\rho}t \in \mathsf{ST}$.

(μ-E) If $r^{\mu\alpha\rho} \in \mathsf{ST}$, $\mathsf{map}_{\lambda\alpha\rho} \in \mathsf{ST}$ and $s^{\rho[\alpha:=\sigma]\to\sigma} \in \mathsf{ST}$, then $rs \in \mathsf{ST}$.

(μ-E$^+$) If $r^{\mu\alpha\rho} \in \mathsf{ST}$, $\mathsf{map}_{\lambda\alpha\rho} \in \mathsf{ST}$ and $s^{\rho[\alpha:=\mu\alpha\rho\times\sigma]\to\sigma} \in \mathsf{ST}$ then $rs \in \mathsf{ST}$.

[8] This is the only embedding in this line of embeddings where types are changed.

This definition is again well-parsed. Hence, we may define functions on ST by recursion on the definition.
Define the function $-' : \mathsf{ST} \to \Lambda'$ by recursion on ST and simultaneously prove that for $r : \rho$ we have $r' : \rho$ and $\mathsf{FTV}(r') = \mathsf{FTV}(r)$ and $\mathsf{FV}(r') = \mathsf{FV}(r)$ as follows:

(eF) The homomorphic term rules for eF.

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

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

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

(The proofs are trivial.) The rule (μ-I) is called the homomorphic μ-introduction rule.

Lemma 4.13 Every (not necessarily proper) subterm r of $\mathsf{map}_{\lambda\alpha\rho}$ is in ST.

Proof Main induction on $\mu\alpha\rho$ along $\succ$ coming from our assumption (S). Side induction on r (i. e., on the definition of Λ). All the term formation rules of eF and (μ-I) are dealt with by the side induction hypothesis. E. g. ($\to$-E): Let $r\mathrm{E}_\to s$ be a subterm of $\mathsf{map}_{\lambda\alpha\rho}$. Then r and s are subterms of $\mathsf{map}_{\lambda\alpha\rho}$. By the side induction hypothesis, r and s are in ST. By the rule (eF) of the definition of ST, $rs \in \mathsf{ST}$. We come to the crucial μ-elimination cases: Let $r^{\mu\alpha'\rho'}s$ be a subterm of $\mathsf{map}_{\lambda\alpha\rho}$. By assumption (S) we have $\mu\alpha\rho \succ \mu\alpha'\rho'$. By the main induction hypothesis for $\mathsf{map}_{\lambda\alpha'\rho'}$ itself, we know that $\mathsf{map}_{\lambda\alpha'\rho'} \in \mathsf{ST}$. r and s are subterms of $\mathsf{map}_{\lambda\alpha\rho}$. By the side induction hypothesis, $r, s \in \mathsf{ST}$. By the definition of ST, $r^{\mu\alpha'\rho'}s \in \mathsf{ST}$. □

Corollary 4.14 If $r, s \in \mathsf{ST}$ and $r^{\mu\alpha\rho}s \in \Lambda$, then $rs \in \mathsf{ST}$. □

Corollary 4.15 $\mathsf{ST} = \Lambda$.

Proof ST has all the closure properties which define Λ. □

Therefore, $-'$ is defined on all terms. Obviously, it is injective on terms.
Again $\mathsf{ST} = \Lambda$ is not only a consequence of (S), but in fact equivalent to (S) (relative to the setting without (U) and (S)) as can be seen by using ordinals as was sketched for the embedding of ESMIT in EMIT.
The statement of Lemma 4.7 is also true for this embedding.

Lemma 4.16 $(r[\vec{x}^{\vec{\rho}} := \vec{s}^{\vec{\rho}}])' = r'[\vec{x}^{\vec{\rho}} := \vec{s}']$ and $(r[\vec{\alpha} := \vec{\sigma}])' = r'[\vec{\alpha} := \vec{\sigma}]$.

Proof Induction on $r \in \mathsf{ST}$: All the cases except (μ-E) and (μ-E$^+$) are straightforward. Consider e. g. (μ-E):

$$((r\mathrm{E}_\mu s)[\vec{x}^{\vec{\rho}} := \vec{s}])' = (r[\vec{x}^{\vec{\rho}} := \vec{s}])'\mathrm{E}_\mu(\mathsf{map}_{\lambda\alpha\rho})'(s[\vec{x}^{\vec{\rho}} := \vec{s}])'.$$

$$(r\mathrm{E}_\mu s)'[\vec{x}^{\vec{\rho}} := \vec{s}'] = r'[\vec{x}^{\vec{\rho}} := \vec{s}'](\mathsf{map}_{\lambda\alpha\rho})'[\vec{x}^{\vec{\rho}} := \vec{s}']s'[\vec{x}^{\vec{\rho}} := \vec{s}'].$$

By assumption $\mathsf{map}_{\lambda\alpha\rho}$ is closed, hence $(\mathsf{map}_{\lambda\alpha\rho})'$ is closed as was proved simultaneously with the definition of $-'$. Hence we are done by the induction hypothesis for r and s. (This part of the lemma did not need the induction on ST. Induction on $r \in \Lambda$ would have been sufficient.)

$$((r\mathrm{E}_\mu s)[\vec{\alpha} := \vec{\sigma}])' = (r[\vec{\alpha} := \vec{\sigma}])'\mathrm{E}_\mu(\mathsf{map}_{(\lambda\alpha\rho)[\vec{\alpha}:=\vec{\sigma}]})'(s[\vec{\alpha} := \vec{\sigma}])'.$$

$$(r\mathrm{E}_\mu s)'[\vec{\alpha} := \vec{\sigma}] = r'[\vec{\alpha} := \vec{\sigma}]\mathrm{E}_\mu(\mathsf{map}_{\lambda\alpha\rho})'[\vec{\alpha} := \vec{\sigma}]s'[\vec{\alpha} := \vec{\sigma}].$$

By induction hypothesis $(\mathsf{map}_{\lambda\alpha\rho}[\vec{\alpha} := \vec{\sigma}])' = (\mathsf{map}_{\lambda\alpha\rho})'[\vec{\alpha} := \vec{\sigma}]$. Because of the induction hypothesis for r and s we need to show $(\mathsf{map}_{(\lambda\alpha\rho)[\vec{\alpha}:=\vec{\sigma}]})' = (\mathsf{map}_{\lambda\alpha\rho}[\vec{\alpha} := \vec{\sigma}])'$. Due to the injectivity of $-'$ this is equivalent to (U) which we assumed in the definition of ISMIT. The case (μ-E$^+$) differs only in the name of the elimination symbol. □

Lemma 4.17 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 Lemma 4.16. The cases $(\beta_\times)$ and (β_+) are obvious. The cases (β_μ) and (β_μ^+) are easy. Check e. g.

$$\begin{aligned}((C_{\mu\alpha\rho}t)\mathrm{E}_\mu\sigma s)' &= (C_{\mu\alpha\rho}t')\mathrm{E}_\mu\sigma(\mathsf{map}_{\lambda\alpha\rho})'s' \\ &\mapsto s'\Big((\mathsf{map}_{\lambda\alpha\rho})'(\mu\alpha\rho)\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma(\mathsf{map}_{\lambda\alpha\rho})'s')t'\Big) \\ &= \Big(s\Big(\mathsf{map}_{\lambda\alpha\rho}(\mu\alpha\rho)\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma\mathsf{map}_{\lambda\alpha\rho}s)t\Big)\Big)'\end{aligned}$$

The other clauses are easily proved by using the induction hypothesis. $\square$

This implies that $-'$ is an embedding of ISMIT in IMIT.

4.7 The variant varIMIT of IMIT

In this section two extensions of IMIT in the spirit of the extension of EMIT to varEMIT are presented. The first one shows that minor modifications may lead to non-normalizing systems. The second one will get the name varIMIT.

4.7.1 Definition

First we study the modification of IMIT following the idea of forming varEMIT. After showing that it is not normalizing we define varIMIT.
The reduction rule (β_μ) of system IMIT uses the term m which comes with the rule (μ-E) only with first type argument $\mu\alpha\rho$ and second type argument σ. For (β_μ^+) the two arguments are $\mu\alpha\rho$ and $\mu\alpha\rho \times \sigma$. In order to get more terms we define a variant of IMIT by considering terms m which are already instantiated to the first type $\mu\alpha\rho$ in the same way as varEMIT was derived from EMIT. In the following only changes to IMIT are given.
Changed clauses in the definition of terms:

(μ-E) If $r^{\mu\alpha\rho} \in \Lambda$, $m^{\forall\alpha.(\mu\alpha\rho\to\alpha)\to\rho[\alpha:=\mu\alpha\rho]\to\rho} \in \Lambda$ and $s^{\rho[\alpha:=\sigma]\to\sigma} \in \Lambda$, then $(rms)^\sigma \in \Lambda$ (also written as $r\mathrm{E}_\mu\sigma ms$ or shorter $r\mathrm{E}_\mu ms$) and $\mathsf{FV}(rs) := \mathsf{FV}(r) \cup \mathsf{FV}(m) \cup \mathsf{FV}(s)$.

(μ-E$^+$) If $r^{\mu\alpha\rho} \in \Lambda$, $m^{\forall\alpha.(\mu\alpha\rho\to\alpha)\to\rho[\alpha:=\mu\alpha\rho]\to\rho} \in \Lambda$ and $s^{\rho[\alpha:=\mu\alpha\rho\times\sigma]\to\sigma} \in \Lambda$ then $(rms)^\sigma \in \Lambda$ (also written as $r\mathrm{E}_\mu^+\sigma ms$ or shorter $r\mathrm{E}_\mu^+ ms$) and $\mathsf{FV}(rs) := \mathsf{FV}(r) \cup \mathsf{FV}(m) \cup \mathsf{FV}(s)$.

Changed clauses in the definition of reduction:

a non-normalizing system

$$\begin{aligned}(\beta_\mu) &\qquad (C_{\mu\alpha\rho}t)\mathrm{E}_\mu\sigma ms \mapsto s\Big(m\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma ms)t\Big) \\ (\beta_\mu^+) &\qquad (C_{\mu\alpha\rho}t)\mathrm{E}_\mu^+\sigma ms \mapsto s\Big(m(\mu\alpha\rho\times\sigma)\Big(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma ms)x\rangle\Big)t\Big)\end{aligned}$$

Let $u \in \mathsf{Vars}$. Set $m := \Lambda\alpha\lambda z^{\mu\alpha\alpha\to\alpha}\lambda u^{\mu\alpha\alpha}.z(C_{\mu\alpha\alpha}u)$. Then for every term $t : \mu\alpha\alpha$ and every type σ we have:

$$\begin{aligned}(C_{\mu\alpha\alpha}t)\mathrm{E}_\mu\sigma m(\lambda y^\sigma y) &\mapsto (\lambda y^\sigma y)(m\sigma(\lambda x^{\mu\alpha\alpha}.x\mathrm{E}_\mu\sigma m(\lambda y^\sigma y))t) \\ &\mapsto m\sigma(\lambda x^{\mu\alpha\alpha}.x\mathrm{E}_\mu\sigma m(\lambda y^\sigma y))t \\ &\to (\lambda z^{\mu\alpha\alpha\to\sigma}\lambda u^{\mu\alpha\alpha}.z(C_{\mu\alpha\alpha}u))(\lambda x^{\mu\alpha\alpha}.x\mathrm{E}_\mu\sigma m(\lambda y^\sigma y))t \\ &\to (\lambda u^{\mu\alpha\alpha}.(\lambda x^{\mu\alpha\alpha}.x\mathrm{E}_\mu\sigma m(\lambda y^\sigma y))(C_{\mu\alpha\alpha}u))t \\ &\to (\lambda x^{\mu\alpha\alpha}.x\mathrm{E}_\mu\sigma m(\lambda y^\sigma y))(C_{\mu\alpha\alpha}t) \\ &\to (C_{\mu\alpha\alpha}t)\mathrm{E}_\mu\sigma m(\lambda y^\sigma y)\end{aligned}$$

Therefore by Lemma 2.50 $(C_{\mu\alpha\alpha}t)\mathrm{E}_\mu\sigma m(\lambda y^\sigma y) \notin \mathsf{sn}$ and it is clear that the term is also not in wn. Note that this does not show the inconsistency of the system, i. e., it does not give a closed term of type $\mu\alpha\alpha$ (which in turn would give a closed term of type 0 and hence closed terms of all types).

We pursue the other possibility of getting more terms by specializing the terms m and define the variant varIMIT of IMIT by considering terms m which are already instantiated to the second type argument. This would not make sense for EMIT because there m is part of the introduction rule.
In the following only changes to IMIT are given. All the statements on IMIT hold as well in varIMIT.
Changed clauses in the definition of terms (always assuming that $\alpha \notin \mathsf{FV}(\sigma)$):

(μ-E) If $r^{\mu\alpha\rho} \in \Lambda$, $m^{\forall\alpha.(\alpha\to\sigma)\to\rho\to\rho[\alpha:=\sigma]} \in \Lambda$ (for $\alpha \notin \mathsf{FV}(\sigma)$) and $s^{\rho[\alpha:=\sigma]\to\sigma} \in \Lambda$, then $(rms)^\sigma \in \Lambda$ (also written as $r\mathrm{E}_\mu\sigma ms$ or shorter $r\mathrm{E}_\mu ms$) and $\mathsf{FV}(rs) := \mathsf{FV}(r) \cup \mathsf{FV}(m) \cup \mathsf{FV}(s)$.

(μ-E$^+$) If $r^{\mu\alpha\rho} \in \Lambda$, $m^{\forall\alpha.(\alpha\to\mu\alpha\rho\times\sigma)\to\rho\to\rho[\alpha:=\mu\alpha\rho\times\sigma]} \in \Lambda$ (for $\alpha \notin \mathsf{FV}(\sigma)$) and $s^{\rho[\alpha:=\mu\alpha\rho\times\sigma]\to\sigma} \in \Lambda$ then $(rms)^\sigma \in \Lambda$ (also written as $r\mathrm{E}_\mu^+\sigma ms$ or shorter $r\mathrm{E}_\mu^+ ms$) and $\mathsf{FV}(rs) := \mathsf{FV}(r) \cup \mathsf{FV}(m) \cup \mathsf{FV}(s)$.

Changed clauses in the definition of reduction:

varIMIT

$$\begin{aligned}(\beta_\mu) &\qquad (C_{\mu\alpha\rho}t)\mathrm{E}_\mu\sigma ms \mapsto s\Big(m(\mu\alpha\rho)(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma ms)t\Big) \\ (\beta_\mu^+) &\qquad (C_{\mu\alpha\rho}t)\mathrm{E}_\mu^+\sigma ms \mapsto s\Big(m(\mu\alpha\rho)\Big(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma ms)x\rangle\Big)t\Big)\end{aligned}$$

We again require $x^{\mu\alpha\rho} \notin \mathsf{FV}(m) \cup \mathsf{FV}(s)$. All the other definitions may be adapted very easily from those in section 4.1.1 on EMIT.
Note that obviously the encodings of IMIT in IMIT–rec and in IMIT–it in section 4.5 cannot be extended to varIMIT because the types of m and m' do not fit.

4.7.2 Embedding IMIT in varIMIT

An embedding $-'$ of IMIT in varIMIT is straightforward to define: Set $\rho' := \rho$ and for every term r^ρ of system IMIT define the term r' of system varIMIT by recursion on r and simultaneously prove that $r' : \rho$ and $\mathsf{FTV}(r') = \mathsf{FTV}(r)$ and $\mathsf{FV}(r') = \mathsf{FV}(r)$ as follows:

(eF) The homomorphic term rules for eF.

(μ-I) The homomorphic μ-introduction rule.

(μ-E) $(r^{\mu\alpha\rho}\mathrm{E}_\mu\sigma ms)' := r'\mathrm{E}_\mu\sigma(\Lambda\alpha.m'\alpha\sigma)s'$.

(μ-E$^+$) $(r^{\mu\alpha\rho}\mathrm{E}_\mu^+\sigma ms)' := r'\mathrm{E}_\mu^+\sigma(\Lambda\alpha.m'\alpha(\mu\alpha\rho\times\sigma))s'$.

In both μ-elimination rules we assume that $\alpha \notin \mathsf{FV}(\sigma) \cup \mathsf{FTV}(m)$. (The proofs are obvious.) The statements of Lemma 4.16 and Lemma 4.17 are true for this definition and are proved by induction on r resp. by induction on $r \to \hat{r}$ which does not make any difficulties.

Chapter 5

Systems with positive inductive types

We define the systems of positive inductive types by constructing their monotonicity witnesses. Because we have even non-strict positivity also witnesses for antitone type dependency have to be defined. In the case of non-interleaving positive inductive types no reference to $\mu\alpha\rho$ is needed in the definitions. A lot of examples (including Gödel's system T) is given to illustrate what can be done with positive inductive types. MePIT and coMePIT are very special instances of NISPIT+ex and NIPIT which serve for purposes of embedding.

5.1 The term rewrite systems **PITit** and **PITrec**

Usually (e. g. in PIT) the canonical witnesses use only iteration. In the system PITrec we enforce the use of primitive recursion instead of iteration. Among other complications we do not get normal forms any more which is achieved in PIT due to careful definitions.

5.1.1 Definition of **PITit** (also called **PIT**)

Define the term rewrite system PITit (also called system PIT) of positive inductive types as one of the systems ESMITit. Let MTypes:=PosTypes. Lemma 3.2 and the definition itself guarantee that MTypes fulfills the required closure conditions.
We have to define $\mathsf{map}_{\lambda\alpha\rho}$ as a closed term of type $\forall\alpha\forall\beta.(\alpha\to\beta)\to\rho\to\rho[\alpha:=\beta]$ for every $\mu\alpha\rho\in\mathsf{MTypes}$, i.e. $\rho\in\mathsf{PosTypes}$ and $\alpha\in\mathsf{Pos}(\rho)$. Because positivity and negativity are defined simultaneously, we define auxiliary terms $\mathsf{comap}_{\lambda\alpha\rho}$ for $\alpha\in\mathsf{Neg}(\rho)$ simultaneously:
Define for every type $\rho\in\mathsf{PosTypes}$ and every $\alpha\in\mathsf{Pos}(\rho)$ (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 $\alpha\in\mathsf{Neg}(\rho)$ (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 $\mathsf{d}(\lambda\alpha\rho)$.
In order to get terms in NF and not too much notation we will set (extending the naming convention for types by π)

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

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

Here we assume that $f\in\mathsf{Vars}$. Moreover, $\underline{\mathsf{map}}_{\lambda\alpha\rho,\pi,\sigma}$ 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 we use as shorthands

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

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

In the definitions we do not write the types of the variables f and x because the only allowed free variables are meant. We shorten $\rho[\alpha:=\sigma]$ to $\rho[\sigma]$ for every ρ and σ.
(triv) Case $\alpha\notin\mathsf{FV}(\rho)$. Then $\alpha\in\mathsf{Pos}(\rho)\cap\mathsf{Neg}(\rho)$. 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"[1].
(V) Case $\rho=\alpha$. Then $\alpha\in\mathsf{Pos}(\rho)\setminus\mathsf{Neg}(\rho)$. Set $\underline{\mathsf{map}}_{\lambda\alpha\rho,\pi,\sigma}:=fx$.[2]
($\to$) Case $\rho=\rho_1\to\rho_2$. If $\alpha\in\mathsf{Pos}(\rho)=\mathsf{Pos}(\rho_2)\cap\mathsf{Neg}(\rho_1)$ then set

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

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

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

Because $\mathsf{d}(\lambda\alpha\rho_1)$ and $\mathsf{d}(\lambda\alpha\rho_2)$ are smaller than $\mathsf{d}(\lambda\alpha\rho)$ this is a correct definition.
($\forall$) Case $\rho=\forall\gamma\tau$. Because of $\alpha\in\mathsf{FV}(\rho)$, $\alpha\neq\gamma$. Therefore, if $\alpha\in\mathsf{Pos}(\rho)$ then $\alpha\in\mathsf{Pos}(\tau)$. Set

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

If $\alpha\in\mathsf{Neg}(\rho)$ then $\alpha\in\mathsf{Neg}(\tau)$. 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 $(\forall\gamma\tau)[\alpha:=\pi]=\forall\gamma.\tau[\alpha:=\pi]$. Also note that according to the definition $\mathsf{d}(\lambda\alpha\tau)<\mathsf{d}(\lambda\alpha\rho)$.
($\times$) Case $\rho=\rho_1\times\rho_2$. If $\alpha\in\mathsf{Pos}(\rho)=\mathsf{Pos}(\rho_1)\cap\mathsf{Pos}(\rho_2)$ then set

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

[1] Typing reasons rule out this setting if $\alpha\in\mathsf{FV}(\rho)$. Only because of this clause it is possible that f does not occur in $\underline{\mathsf{map}}_{\lambda\alpha\rho,\pi,\sigma}$ or $\underline{\mathsf{comap}}_{\lambda\alpha\rho,\pi,\sigma}$, and which would otherwise be λI-terms, i. e., terms where only variables actually occurring free are abstracted.
The definition in [Lei90, p. 311] does not make this case distinction but uses (triv) only for the case where ρ is a variable different from α. It seems that the case (μ) then makes the definition unfounded. The definition in [How92, p. 27] has (triv) but no comap because it is designed for strictly positive inductive types.
[2] Therefore, $\mathsf{map}_{\lambda\alpha\alpha}$ will not be η-normal (The rule $\lambda x^{\alpha}.fx\mapsto_{\eta_\to}f$ would apply).

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

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

(+) Case $\rho = \rho_1 + \rho_2$. If $\alpha \in \mathsf{Pos}(\rho) = \mathsf{Pos}(\rho_1) \cap \mathsf{Pos}(\rho_2)$ then set

$$\underline{\mathsf{map}}_{\lambda\alpha\rho,\pi,\sigma} := x\mathrm{E}_+ \Big(\lambda y_1^{\rho_1[\pi]}.\mathsf{INL}_{\rho_2[\sigma]} \underline{\mathsf{map}}_{\lambda\alpha\rho_1,\pi,\sigma}[f, y_1]\Big) \Big(\lambda y_2^{\rho_2[\pi]}.\mathsf{INR}_{\rho_1[\sigma]} \underline{\mathsf{map}}_{\lambda\alpha\rho_2,\pi,\sigma}[f, y_2]\Big).$$

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

$$\underline{\mathsf{comap}}_{\lambda\alpha\rho,\pi,\sigma} := x\mathrm{E}_+ \Big(\lambda y_1^{\rho_1[\sigma]}.\mathsf{INL}_{\rho_2[\pi]} \underline{\mathsf{comap}}_{\lambda\alpha\rho_1,\pi,\sigma}[f, y_1]\Big) \Big(\lambda y_2^{\rho_2[\sigma]}.\mathsf{INR}_{\rho_1[\pi]} \underline{\mathsf{comap}}_{\lambda\alpha\rho_2,\pi,\sigma}[f, y_2]\Big).$$

(μ) Case $\rho = \mu\gamma\tau$. Because of $\alpha \in \mathsf{FV}(\rho)$, $\alpha \neq \gamma$. If $\alpha \in \mathsf{Pos}(\rho) = \mathsf{Pos}(\tau)$ then set

$$\underline{\mathsf{map}}_{\lambda\alpha\rho,\pi,\sigma} := x\mathrm{E}_\mu \rho[\sigma] \Big(\lambda y^{\tau[\pi][\gamma := \mu\gamma.\tau[\sigma]]}.C_{\mu\gamma.\tau[\sigma]}(\underline{\mathsf{map}}_{\lambda\alpha.\tau[\gamma := \mu\gamma.\tau[\sigma]],\pi,\sigma}[f, y])\Big).$$

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

$$\underline{\mathsf{comap}}_{\lambda\alpha\rho,\pi,\sigma} := x\mathrm{E}_\mu \rho[\pi] \Big(\lambda y^{\tau[\sigma][\gamma := \mu\gamma.\tau[\pi]]}.C_{\mu\gamma\tau[\pi]}(\underline{\mathsf{comap}}_{\lambda\alpha.\tau[\gamma := \mu\gamma.\tau[\pi]],\pi,\sigma}[f, y])\Big).$$

Let us verify that this case (μ) is a correct definition clause. We may assume that $\alpha, \gamma \notin \mathsf{FV}(\pi) \cup \mathsf{FV}(\sigma)$. Therefore $\rho[\pi] = \mu\gamma.\tau[\pi]$ and $\rho[\sigma] = \mu\gamma.\tau[\sigma]$ and

$$(\tau[\gamma := \mu\gamma.\tau[\rho_1]])[\rho_2] = \tau[\rho_2][\gamma := \mu\gamma.\tau[\rho_1]]$$

for any $\rho_1, \rho_2 \in \{\pi, \sigma\}$ (use Corollary 2.2, Lemma 2.3 and Lemma 2.5). Let $\rho_0 \in \{\pi, \sigma\}$. $\rho \in \mathsf{PosTypes}$, hence by Lemma 3.2 $\rho[\rho_0] \in \mathsf{PosTypes}$. Therefore $\mu\gamma.\tau[\rho_0] \in \mathsf{PosTypes}$ and consequently also $\tau[\gamma := \mu\gamma.\tau[\rho_0]] \in \mathsf{PosTypes}$. Because $\alpha \notin \mathsf{FV}(\mu\gamma.\tau[\rho_0])$, Lemma 3.2 also gives us for $\alpha \in \mathsf{Pos}(\tau)$ that $\alpha \in \mathsf{Pos}(\tau[\gamma := \mu\gamma.\tau[\rho_0]])$ and for $\alpha \in \mathsf{Neg}(\tau)$ that $\alpha \in \mathsf{Neg}(\tau[\gamma := \mu\gamma.\tau[\rho_0]])$. By Lemma 3.6 (2.) $\mathsf{d}(\lambda\alpha.\tau[\gamma := \mu\gamma.\tau[\rho_0]]) < \mathsf{d}(\lambda\alpha\rho)$. Therefore the induction hypothesis applies in both cases, and the terms are well-formed.
It is obvious that the term generation rule (μ-E^+) is not used in the definition of map and comap.

Lemma 5.1 The terms $\mathsf{map}_{\lambda\alpha\rho}$ for $\mu\alpha\rho \in \mathsf{PosTypes}$ satisfy the conditions (C), (U) and (S) of ESMIT.

Proof Of course, we have to prove appropriate statements also for $\mathsf{comap}_{\lambda\alpha\rho}$. They are (co-C) which does not differ from (C) in the verbal representation and (co-U) which is derived from (U) by replacing "map" by "comap". (S) has to be completely rewritten to (new-S):
There is a well-founded relation $\succ$ on abstracted types of the form $\lambda\alpha\rho$ such that for every $\rho \in \mathsf{PosTypes}$ and $\alpha \in \mathsf{Pos}(\rho)$ and every subterm $C_{\mu\alpha'\rho'}t$ of $\mathsf{map}_{\lambda\alpha\rho}$ we have $\lambda\alpha\rho \succ \lambda\alpha'\rho'$ and for every $\rho \in \mathsf{PosTypes}$ and $\alpha \in \mathsf{Neg}(\rho)$ and every subterm $C_{\mu\alpha'\rho'}t$ of $\mathsf{comap}_{\lambda\alpha\rho}$ we also have $\lambda\alpha\rho \succ \lambda\alpha'\rho'$.
(C) and (co-C): This is clear because in the definitions of $\underline{\mathsf{map}}$ and $\underline{\mathsf{comap}}$ the variable α always only occurs bound. (U) and (co-U): Applying the type substitution does not lead to a jump from one case in the definition to another: neither in the distinction whether $\alpha \in \mathsf{FV}(\rho)$ nor in the analysis of the outermost type constructor nor in the cases due to positivity or negativity.
(new-S): The following approach does not work. Set

$$\lambda\alpha\rho \succ \lambda\alpha'\rho' :\Leftrightarrow \mathsf{d}(\lambda\alpha\rho) > \mathsf{d}(\lambda\alpha'\rho').$$

In the (μ)-clause we have to check

$$\mathsf{d}(\lambda\alpha\mu\gamma\tau) > \mathsf{d}(\lambda\gamma.\tau[\alpha := \sigma]) = \mathsf{d}(\lambda\gamma\tau)$$

(assuming $\gamma \notin \mathsf{FV}(\sigma)$). This is not true in general (e. g. for $\tau := \alpha \times \forall\beta\forall\delta\gamma$ assuming that the variables are different). The definition of h just aimed at solving this problem. Setting

$$\lambda\alpha\rho \succ \lambda\alpha'\rho' :\Leftrightarrow \mathsf{h}(\lambda\alpha\rho) > \mathsf{h}(\lambda\alpha'\rho'),$$

it is easy to show (new-S) by induction on $\mathsf{d}(\lambda\alpha\rho)$ (or on the definition of map and comap) using Lemma 3.5. It is also easy to see that $\mathsf{map}_{\lambda\alpha\rho}$ and $\mathsf{comap}_{\lambda\alpha\rho}$ could have been defined by recursion on $\mathsf{h}(\lambda\alpha\rho)$. □

Note that it would be nearly the same proof for showing that PIT is also one of the systems ISMIT (due to the observation that in the rule (μ) of the definition of $\underline{\mathsf{map}}$ and $\underline{\mathsf{comap}}$ the μ-introduction and the μ-elimination happen with μ-types of the same height and depth).

Lemma 5.2 map and comap are always terms in NF.

Proof By induction on the definition using the statements of Lemma 2.25 and Lemma 2.26 which hold for ESMIT. □

Therefore, by Lemma 4.7 for every term r of PIT in NF, the term r' in EMIT, to which r is mapped by the canonical embedding $-'$ defined in section 4.2.2, is also in NF.
Note that the rule (μ-E^+) is not used for the definition of map and comap. Note also that the first argument to $\underline{\mathsf{map}}$ and $\underline{\mathsf{comap}}$ is always the variable f.

5.1.2 Examples of term rewriting in PIT

In the following some of the examples of inductive types introduced in section 3.1 which are types in PIT are studied more carefully. The types of the variables which are bound will always be omitted except at the binder.
As a first example consider $\mathsf{nat} = \mu\alpha.1 + \alpha$. It is immediate that

$$\mathsf{map}_{\lambda\alpha.1+\alpha} = \Lambda\alpha\Lambda\beta\lambda f^{\alpha\to\beta}\lambda x^{1+\alpha}.x\mathrm{E}_+(\lambda y^1.\mathsf{INL}_\beta y)(\lambda y^\alpha.\mathsf{INR}_1(fy)).$$

Define

$$\begin{array}{rcl} 0 & := & C_{\mathsf{nat}}(\mathsf{INL}_{\mathsf{nat}}\mathsf{IN1}) : \mathsf{nat} \\ \mathsf{S} & := & \lambda x^{\mathsf{nat}}.C_{\mathsf{nat}}(\mathsf{INR}_1 x) : \mathsf{nat} \to \mathsf{nat} \end{array}$$

Given terms $a : \sigma$ and $b : \mathsf{nat} \to \sigma \to \sigma$ we define $F_{a,b} := \lambda x^{\mathsf{nat}}.x\mathrm{E}_\mu^+ \sigma s : \mathsf{nat} \to \sigma$ with

$$s := \lambda x^{1+\mathsf{nat}\times\sigma}.x\mathrm{E}_+(\lambda y^1.a)(\lambda y^{\mathsf{nat}\times\sigma}.b(y\mathsf{L})(y\mathsf{R})),$$

where $x, y \notin \mathsf{FV}(a) \cup \mathsf{FV}(b)$. We get the following reduction behaviour: $F_{a,b}0 \to^+ a$ and $F_{a,b}(\mathsf{S}t) \to^+ bt(F_{a,b}t)$. Therefore we may say that Gödel's system T (see e. g. [GLT89]) is a subsystem of PIT.

Consider $\mathsf{tree} = \mathsf{tree}(\mathsf{nat}) = \mu\alpha.1 + (\mathsf{nat} \to \alpha)$. We see that

$$\mathsf{map}_{\lambda\alpha.1+(\mathsf{nat}\to\alpha)} = \Lambda\alpha\Lambda\beta\lambda f^{\alpha\to\beta}\lambda x^{1+(\mathsf{nat}\to\alpha)}.x\mathrm{E}_+(\lambda y^1.\mathsf{INL}_{\mathsf{nat}\to\beta}y)(\lambda y^{\mathsf{nat}\to\alpha}.\mathsf{INR}_1(\lambda z^{\mathsf{nat}}.f(yz))).$$

Define

$$\begin{array}{lll} \mathsf{nil} & := & C_{\mathsf{tree}}(\mathsf{INL}_{\mathsf{nat}\to\mathsf{tree}}\mathsf{IN1}) : \mathsf{tree} \\ \mathsf{lim} & := & \lambda x^{\mathsf{nat}\to\mathsf{tree}}.C_{\mathsf{tree}}(\mathsf{INR}_1 x) : (\mathsf{nat}\to\mathsf{tree})\to\mathsf{tree} \end{array}$$

Given terms $a : \sigma$ and $b : (\mathsf{nat} \to \mathsf{tree}) \to (\mathsf{nat} \to \sigma) \to \sigma$ we define the term $F_{a,b} := \lambda x^{\mathsf{tree}}.x\mathrm{E}^+_\mu \sigma s : \mathsf{tree} \to \sigma$ with

$$s := \lambda z^{1+(\mathsf{nat}\to(\mathsf{tree}\times\sigma))}.z\mathrm{E}_+(\lambda y^1.a)(\lambda y^{\mathsf{nat}\to(\mathsf{tree}\times\sigma)}.b(\lambda u^{\mathsf{nat}}.yu\mathsf{L})(\lambda u^{\mathsf{nat}}.yu\mathsf{R})),$$

where $y, z \notin \mathsf{FV}(a) \cup \mathsf{FV}(b)$. The reduction behaviour is as follows: $F_{a,b}\mathsf{nil} \to^+ a$ and

$$F_{a,b}(\mathsf{lim}\ t) \to^+ b(\lambda u^{\mathsf{nat}}.tu)(\lambda u^{\mathsf{nat}}.F_{a,b}(tu)).$$

Although there is the unpleasant $\lambda u^{\mathsf{nat}}.tu$ instead of t we see that this reduction behaviour reflects our intuition of recursion on countably branching trees (and by adding $\eta_\to$-reduction to the reduction rules we could avoid this nuisance without making the normalization proof harder).

We turn to $\mathsf{cont}(\rho) = \mu\alpha.1 + ((\alpha \to \rho) \to \rho)$ with $\rho \in \mathsf{PosTypes}$. We get

$$\mathsf{map}_{\lambda\alpha.1+((\alpha\to\rho)\to\rho)} = \Lambda\alpha\Lambda\beta\lambda f^{\alpha\to\beta}\lambda x^{1+((\alpha\to\rho)\to\rho)}.x\mathrm{E}_+(\lambda y^1.\mathsf{INL}_{(\beta\to\rho)\to\rho}y)s$$

with

$$s := \lambda y^{(\alpha\to\rho)\to\rho}.\mathsf{INR}_1(\lambda g^{\beta\to\rho}.y(\lambda h^\alpha.g(fh))).$$

Define

$$\begin{array}{lll} \mathsf{nil} & := & C_{\mathsf{cont}(\rho)}(\mathsf{INL}_{(\mathsf{cont}(\rho)\to\rho)\to\rho}\mathsf{IN1}) : \mathsf{cont}(\rho) \\ \mathsf{lim} & := & \lambda x^{(\mathsf{cont}(\rho)\to\rho)\to\rho}.C_{\mathsf{cont}(\rho)}(\mathsf{INR}_1 x) : ((\mathsf{cont}(\rho)\to\rho)\to\rho)\to\mathsf{cont}(\rho) \end{array}$$

Let a be a closed term of type ρ. Define a closed term E of type $\mathsf{cont}(\rho) \to \rho$ by

$$E := \lambda x^{\mathsf{cont}(\rho)}.x\mathrm{E}_\mu(\lambda z^{1+((\rho\to\rho)\to\rho)}.z\mathrm{E}_+(\lambda y^1.a)(\lambda y^{(\rho\to\rho)\to\rho}.y(\lambda u^\rho u))).$$

We get the following reduction behaviour: $E\mathsf{nil} \to^+ a$ and

$$E(\mathsf{lim}\ t) \to^+ (\lambda y^{(\rho\to\rho)\to\rho}.y(\lambda u^\rho u))(\lambda g^{\rho\to\rho}.t(\lambda h^{\mathsf{cont}(\rho)}.g(Eh))) \to^+ t(\lambda h^{\mathsf{cont}(\rho)}.Eh).$$

If we ignore the difference between $\lambda h^{\mathsf{cont}(\rho)}.Eh$ and E (which could again easily be removed by introducing η-reduction for $\to$) we may say that the following definition is by iteration on $\mathsf{cont}(\rho)$:

$$\begin{array}{lll} E\mathsf{nil} & := & a \\ E(\mathsf{lim}\ t) & := & tE \end{array}$$

We see that the recursive call is by no means to E with some term smaller than $\mathsf{lim}\ t$ in any sense. The term E which stands for the function about to be defined is even fed in as an argument to the term t. But nevertheless in chapter 9 it will be shown that PIT is strongly normalizing and hence also the lambda calculus extended by this definition when seen as term rewrite rules.
As was mentioned in section 3.1 this type was studied before by Martin Hofmann. It was brought to my attention by Ulrich Berger in the unpublished manuscript [Ber95]. The possibility of defining a function with clauses as above and having termination guaranteed because it is simply an instance of iteration on some inductively defined set caught my attention to an extent which made this thesis possible.

Finally consider $\mathsf{Tree} = \mu\alpha.1 + (\mathsf{tree}(\alpha) \to \alpha)$. By unwinding the definitions we get

$$\mathsf{map}_{\lambda\alpha.1+(\mathsf{tree}(\alpha)\to\alpha)} = \Lambda\alpha\Lambda\beta\lambda f^{\alpha\to\beta}\lambda x^{1+(\mathsf{tree}(\alpha)\to\alpha)}.x\mathrm{E}_+(\lambda y^1.\mathsf{INL}_{\mathsf{tree}(\beta)\to\beta}y)s,$$

where we set

$$s := \lambda y^{\mathsf{tree}(\alpha)\to\alpha}.\mathsf{INR}_1(\lambda z^{\mathsf{tree}(\beta)}.f(y(z\mathrm{E}_\mu(\lambda i^{1+(\beta\to\mathsf{tree}(\alpha))}.C_{\mathsf{tree}(\alpha)}t)))),$$

where we set

$$t := i\mathrm{E}_+(\lambda k^1.\mathsf{INL}_{\alpha\to\mathsf{tree}(\alpha)}k)(\lambda k^{\beta\to\mathsf{tree}(\alpha)}.\mathsf{INR}_1(\lambda m^\alpha.k(fm))).$$

Define

$$\begin{array}{lll} \mathsf{nil} & := & C_{\mathsf{Tree}}(\mathsf{INL}_{\mathsf{tree}(\mathsf{Tree})\to\mathsf{Tree}}\mathsf{IN1}) : \mathsf{Tree} \\ \mathsf{lim} & := & \lambda x^{\mathsf{tree}(\mathsf{Tree})\to\mathsf{Tree}}.C_{\mathsf{Tree}}(\mathsf{INR}_1 x) : (\mathsf{tree}(\mathsf{Tree})\to\mathsf{Tree})\to\mathsf{Tree} \end{array}$$

Given terms $a : \sigma$ and $b : (\mathsf{tree}(\sigma) \to \sigma) \to \sigma$ we define $F_{a,b} := \lambda x^{\mathsf{Tree}}.x\mathrm{E}_\mu \sigma s : \mathsf{Tree} \to \sigma$ with

$$s := \lambda x^{1+(\mathsf{tree}(\sigma)\to\sigma)}.x\mathrm{E}_+(\lambda y^1.a)b,$$

where $x, y \notin \mathsf{FV}(a)$ and $x \notin \mathsf{FV}(b)$. We get the following reduction behaviour: $F_{a,b}\mathsf{nil} \to^+ a$ and

$$F_{a,b}(\mathsf{lim}\ t) \to^+ b(\lambda z^{\mathsf{tree}(\sigma)}.F_{a,b}(t(z\mathrm{E}_\mu(\lambda i^{1+(\sigma\to\mathsf{tree}(\mathsf{Tree}))}.C_{\mathsf{tree}(\mathsf{Tree})}(i\mathrm{E}_+ s_1 s_2))))) \text{ with}$$

$$\begin{array}{lll} s_1 & := & \lambda k^1.\mathsf{INL}_{\mathsf{Tree}\to\mathsf{tree}(\mathsf{Tree})}k \\ s_2 & := & \lambda k^{\sigma\to\mathsf{tree}(\mathsf{Tree})}.\mathsf{INR}_1(\lambda m^{\mathsf{Tree}}.k(F_{a,b}m)). \end{array}$$

Note the occurrence of $F_{a,b}$ in s_2.

5.1.3 Definition of PITrec

We define the system PITrec as one of the systems ESMITrec. As for PIT let MTypes:=PosTypes. The definition of the map-terms only introduces a reference to (μ-E) in the case (μ). As was mentioned in the proof of Lemma 5.1, the definition of $\mathsf{map}_{\lambda\alpha\rho}$ and $\mathsf{comap}_{\lambda\alpha\rho}$ could as well have been done by recursion on $\mathsf{h}(\lambda\alpha\rho)$. For PITrec we define $\mathsf{map}_{\lambda\alpha\rho}$ and $\mathsf{comap}_{\lambda\alpha\rho}$ by recursion on $\mathsf{h}(\lambda\alpha\rho)$ introducing the $\underline{\mathsf{map}}$- and $\underline{\mathsf{comap}}$-terms as for PITit and only change the clause (μ) to:
(μ^+) Case $\rho = \mu\gamma\tau$. Because of $\alpha \in \mathsf{FV}(\rho)$, $\alpha \neq \gamma$. If $\alpha \in \mathsf{Pos}(\rho) = \mathsf{Pos}(\tau)$ then set

$$\underline{\mathsf{map}}_{\lambda\alpha\rho,\pi,\sigma} := x\mathrm{E}^+_\mu \rho[\sigma]\Big(\lambda y^{\tau[\pi][\gamma:=(\mu\gamma.\tau[\pi])\times(\mu\gamma.\tau[\sigma])]}.C_{\mu\gamma.\tau[\sigma]}(\underline{\mathsf{map}}_{\lambda\alpha.\tau[\gamma:=\mu\gamma.\tau[\sigma]],\pi,\sigma}[f,r])\Big) \text{ with}$$

$$r := \underline{\mathsf{map}}_{\lambda\gamma.\tau[\pi],(\mu\gamma.\tau[\pi])\times(\mu\gamma.\tau[\sigma]),\mu\gamma.\tau[\sigma]}[\lambda z^{(\mu\gamma.\tau[\pi])\times(\mu\gamma.\tau[\sigma])}.z\mathsf{R}, y].$$

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

$$\underline{\mathsf{comap}}_{\lambda\alpha\rho,\pi,\sigma} := x\mathrm{E}^+_\mu \rho[\pi]\Big(\lambda y^{\tau[\sigma][\gamma:=(\mu\gamma.\tau[\sigma])\times(\mu\gamma.\tau[\pi])]}.C_{\mu\gamma\tau[\pi]}(\underline{\mathsf{comap}}_{\lambda\alpha.\tau[\gamma:=\mu\gamma.\tau[\pi]],\pi,\sigma}[f,r])\Big) \text{ with}$$

$$r := \underline{\mathsf{map}}_{\lambda\gamma.\tau[\sigma],(\mu\gamma.\tau[\sigma])\times(\mu\gamma.\tau[\pi]),\mu\gamma.\tau[\pi]}[\lambda z^{(\mu\gamma.\tau[\sigma])\times(\mu\gamma.\tau[\pi])}.z\mathsf{R}, y].$$

The justification is the same as for (μ), where d is replaced by h, and by using $\mathsf{h}(\lambda\alpha\mu\gamma\tau) > \mathsf{h}(\mu\gamma.\tau[\alpha := \rho_0])$ for $\rho_0 \in \{\pi, \sigma\}$.

The use of recursion instead of iteration in these clauses clearly follows the idea of the first coding of IMIT in IMIT–it in section 4.5.2. It is mainly intended to show that even the definitional principle has to be improved (from d to h) in order to get the map-terms without iteration.
Lemma 5.1 is also true for PITrec and proved in the same way. Note that the statement of Lemma 5.2 is no longer true:

$$\underline{\mathsf{map}}_{\lambda\alpha\alpha,\rho\times\sigma,\sigma}[\lambda z^{\rho\times\sigma}.z\mathsf{R},x] = (\lambda z^{\rho\times\sigma}.z\mathsf{R})x \notin \mathsf{NF}.$$

Such a term will be used for the build-up of e. g. $\mathsf{map}_{\lambda\alpha\mu\gamma.\alpha+\gamma}$ and by Lemma 2.27 it follows that $\mathsf{map}_{\lambda\alpha\mu\gamma.\alpha+\gamma} \notin \mathsf{NF}$.
In some extensional theory it is possible to show that these map-terms are equal to the ones of PITit. But this is by no means easy and requires functoriality of both monotonicity witnesses (cf. appendix B).

5.2 The term rewrite systems SPIT, PIT+ex and SPIT+ex

Define the term rewrite system SPIT of strictly positive inductive types as the restriction of PITit to SPosTypes, i. e., set MTypes := SPosTypes $\subseteq$ PosTypes (by Lemma 3.4) and check that every construction made in the definition of PITit does not lead out of MTypes. Therefore SPIT is one of the systems ESMITit.
Lemma 3.1 and the definition of SPosTypes show that MTypes fulfills the required closure conditions. Therefore, it is easily shown by induction on $\mathsf{d}(\lambda\alpha\rho)$ that for $\rho \in \mathsf{SPosTypes}$ and $\alpha \in \mathsf{Pos}(\rho)$ the term $\mathsf{map}_{\lambda\alpha\rho}$ of PITit is also a term of SPIT and for $\alpha \in \mathsf{Neg}(\rho)$ the term $\mathsf{comap}_{\lambda\alpha\rho}$ of PITit is also a term of SPIT. (Of course, the respective property has to be proved for $\underline{\mathsf{map}}_{\lambda\alpha\rho,\pi,\sigma}$ and $\underline{\mathsf{comap}}_{\lambda\alpha\rho,\pi,\sigma}$ with $\pi,\sigma \in \mathsf{SPosTypes}$.) Note that the induction hypothesis is always applied to abstracted types of the form $\lambda\alpha\tau$ with $\tau \in \mathsf{SPosTypes}$. The following approach gives more information:

Lemma 5.3 If ρ, π and $\sigma \in \mathsf{SPosTypes}$ and $\alpha \in \mathsf{SPos}(\rho)$, then $\underline{\mathsf{map}}_{\lambda\alpha\rho,\pi\sigma}$ is a term of SPIT.

Proof Induction on $\mathsf{d}(\lambda\alpha\rho)$. The only cases of interest are $(\to)$ and (μ). $(\to)$: Case $\rho = \rho_1 \to \rho_2$. If $\alpha \in \mathsf{SPos}(\rho) = \mathsf{SPos}(\rho_2) \setminus \mathsf{FV}(\rho_1)$, then $\alpha \notin \mathsf{FV}(\rho_1)$. Therefore $\underline{\mathsf{comap}}_{\lambda\alpha\rho_1,\pi,\sigma}[f,y] = y$, hence $\underline{\mathsf{map}}_{\lambda\alpha\rho,\pi,\sigma} = \lambda y^{\rho_1[\sigma]}.\underline{\mathsf{map}}_{\lambda\alpha\rho_2,\pi,\sigma}[f,xy]$. (Note that $x\mathrm{E}_{\to}y$ is well-typed because $\rho_1[\pi] = \rho_1[\sigma]$.) Now the induction hypothesis applies. (μ): Case $\rho = \mu\gamma\tau$. If $\alpha \in \mathsf{SPos}(\rho) = \mathsf{SPos}(\tau)$, then making the assumption that $\alpha,\gamma \notin \mathsf{FV}(\sigma)$ we only have to show that $\alpha \in \mathsf{SPos}(\tau[\gamma := \mu\gamma.\tau[\sigma]])$. This follows from Lemma 3.1 because $\alpha \notin \mathsf{FV}(\mu\gamma.\tau[\sigma])$. □

For easy reference the following list of properties of $\underline{\mathsf{map}}$ is given. It could also serve as a definition (by induction on $\mathsf{d}(\lambda\alpha\rho)$). All the types are assumed to range over SPosTypes.
If $\alpha \notin \mathsf{FV}(\rho)$, then $\underline{\mathsf{map}}_{\lambda\alpha\rho,\pi,\sigma} = x$. All the other cases are under the proviso $\alpha \in \mathsf{SPos}(\rho) \cap \mathsf{FV}(\rho)$, where ρ is the type which is λ-abstracted on the left side.

$$\begin{array}{lcl}
\underline{\mathsf{map}}_{\lambda\alpha\alpha,\pi,\sigma} & = & fx \\
\underline{\mathsf{map}}_{\lambda\alpha.\rho_1\to\rho_2,\pi,\sigma} & = & \lambda y^{\rho_1[\sigma]}.\underline{\mathsf{map}}_{\lambda\alpha\rho_2,\pi,\sigma}[f,xy] \\
\underline{\mathsf{map}}_{\lambda\alpha\forall\gamma\tau,\pi,\sigma} & = & \Lambda\gamma.\underline{\mathsf{map}}_{\lambda\alpha\tau,\pi,\sigma}[f,x\gamma] \\
\underline{\mathsf{map}}_{\lambda\alpha.\rho_1\times\rho_2,\pi,\sigma} & = & \left\langle\underline{\mathsf{map}}_{\lambda\alpha\rho_1,\pi,\sigma}[f,x\mathsf{L}],\underline{\mathsf{map}}_{\lambda\alpha\rho_2,\pi,\sigma}[f,x\mathsf{R}]\right\rangle \\
\underline{\mathsf{map}}_{\lambda\alpha.\rho_1+\rho_2,\pi,\sigma} & = & x\mathrm{E}_+\left(\lambda y_1^{\rho_1[\pi]}.\mathsf{INL}_{\rho_2[\sigma]}\underline{\mathsf{map}}_{\lambda\alpha\rho_1,\pi,\sigma}[f,y_1]\right)\left(\lambda y_2^{\rho_2[\pi]}.\mathsf{INR}_{\rho_1[\sigma]}\underline{\mathsf{map}}_{\lambda\alpha\rho_2,\pi,\sigma}[f,y_2]\right) \\
\underline{\mathsf{map}}_{\lambda\alpha\mu\gamma\tau,\pi,\sigma} & = & x\mathrm{E}_\mu\rho[\sigma]\left(\lambda y^{\tau[\pi][\gamma:=\mu\gamma.\tau[\sigma]]}.C_{\mu\gamma.\tau[\sigma]}(\underline{\mathsf{map}}_{\lambda\alpha.\tau[\gamma:=\mu\gamma.\tau[\sigma]],\pi,\sigma}[f,y])\right)
\end{array}$$

Because of Lemma 5.1 the terms $\mathsf{map}_{\lambda\alpha\rho}$ for $\mu\alpha\rho \in \mathsf{SPosTypes}$ satisfy the conditions (C), (U) and (S) of ESMIT.
If one leaves out the rule (μ-E$^+$) in the definition of terms for SPIT (and consequently also the rule for full primitive recursion) and also removes universal type quantification, one gets a system which is essentially Loader's calculus of inductive types in [Loa97].

The systems PIT+ex and SPIT+ex are defined as systems ESMIT+ex setting MTypes:=PosTypes and MTypes:=SPosTypes, respectively.
The definition of map and comap for PIT is extended by the following clause (recall that the clauses have the proviso stated in the clause (triv)):
($\exists$) Case $\rho = \exists\gamma\tau$. If $\alpha \in \mathsf{Pos}(\rho)$ then $\alpha \in \mathsf{Pos}(\tau)$ (due to the proviso $\alpha \in \mathsf{FV}(\rho)$). Set

$$\underline{\mathsf{map}}_{\lambda\alpha\rho,\pi,\sigma} := x\mathrm{E}_\exists(\Lambda\gamma\lambda y^{\tau[\pi]}.C_{\exists\gamma.\tau[\sigma],\gamma}\underline{\mathsf{map}}_{\lambda\alpha\tau,\pi,\sigma}[f,y]).$$

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

$$\underline{\mathsf{comap}}_{\lambda\alpha\rho,\pi,\sigma} := x\mathrm{E}_\exists(\Lambda\gamma\lambda y^{\tau[\sigma]}.C_{\exists\gamma.\tau[\pi],\gamma}\underline{\mathsf{comap}}_{\lambda\alpha\tau,\pi,\sigma}[f,y]).$$

The proviso for Λ-abstraction is met as we may assume that $\gamma \notin \mathsf{FV}(\pi)\cup\mathsf{FV}(\sigma)$. The justification is the same as for the clause ($\forall$).
Obviously, these clauses may also be interpreted in SPIT+ex. Indeed, the statement of Lemma 5.3 (where SPIT is replaced by SPIT+ex) holds and may be proved in the same way. The clauses for $\underline{\mathsf{comap}}$ are not needed as was the case for SPIT.

5.3 The term rewrite systems NIPIT, NISPIT and NISPIT+ex

We define systems of non-interleaving positive inductive types, hence nesting is allowed but not interleaving using the parameters of a type $\mu\alpha\rho$ for forming another μ-type.
Define NIPIT as one of the systems ESMITit by restricting PIT to the type system defined by MTypes:=NIPosTypes. Due to Lemma 3.10 the condition (sub) is satisfied for NIPosTypes. We have to check that the terms $\mathsf{map}_{\lambda\alpha\rho}$ with $\mu\alpha\rho \in \mathsf{NIPosTypes}$ are also terms in this restricted system.

Lemma 5.4 Let ρ, π and $\sigma \in \mathsf{NIPosTypes}$. If $\alpha \in \mathsf{NIPos}(\rho)$, then $\underline{\mathsf{map}}_{\lambda\alpha\rho,\pi\sigma}$ is a term of NIPIT. If $\alpha \in \mathsf{NINeg}(\rho)$, then $\underline{\mathsf{comap}}_{\lambda\alpha\rho,\pi\sigma}$ is a term of NIPIT.

Proof Induction on ρ. The only case not being completely obvious is (μ): Let $\rho = \mu\gamma\tau$ and $\alpha \in \mathsf{FV}(\rho)$. Therefore $\alpha \notin \mathsf{NIPos}(\tau) \setminus \mathsf{FV}(\rho) = \mathsf{NIPos}(\rho)$ and $\alpha \notin (\mathsf{NINeg}(\tau) \setminus \mathsf{FV}(\rho)) \cup \{\gamma\} = \mathsf{NINeg}(\rho)$. Hence, the case (μ) never applies! □

The proof shows that $\underline{\mathsf{map}}_{\lambda\alpha\rho}$ and $\underline{\mathsf{comap}}_{\lambda\alpha\rho}$ for NIPIT could also be defined by induction on ρ by just leaving out the clause (μ) in the definition of map and comap for PIT. This definition has been given in [Geu92]. It seems that in [Lei90] the definition even in the general case of interleaved μ-types, viz. in PIT, has been given by induction on the type which unfortunately is not possible. Because map and comap are defined without the term rules (μ-I), (μ-E) and (μ-E$^+$), NIPIT is trivially also one of the systems ESMITrec.
Define NISPIT by restricting SPIT to the types MTypes:=NISPosTypes. As for NIPIT and SPIT we get a system ESMITit. The definition of $\mathsf{map}_{\lambda\alpha\rho}$ may be done by induction on ρ and does not contain a specific clause for (μ).

The system NISPIT+ex is defined as the restriction of SPIT+ex to MTypes:=NISPosTypes (including the existential quantifier). We get the following clauses (which are simply those of SPIT+ex without the μ-clause):

$$
\begin{array}{lcl}
\underline{\mathsf{map}}_{\lambda\alpha\rho,\pi,\sigma} & = & x, \text{ if } \alpha \notin \mathsf{FV}(\rho). \text{ The other clauses only otherwise!} \\
\underline{\mathsf{map}}_{\lambda\alpha\alpha,\pi,\sigma} & = & fx \\
\underline{\mathsf{map}}_{\lambda\alpha.\rho_1\to\rho_2,\pi,\sigma} & = & \lambda y^{\rho_1[\sigma]}.\underline{\mathsf{map}}_{\lambda\alpha\rho_2,\pi,\sigma}[f, xy] \\
\underline{\mathsf{map}}_{\lambda\alpha\forall\gamma\tau,\pi,\sigma} & = & \Lambda\gamma.\underline{\mathsf{map}}_{\lambda\alpha\tau,\pi,\sigma}[f, x\gamma] \\
\underline{\mathsf{map}}_{\lambda\alpha.\rho_1\times\rho_2,\pi,\sigma} & = & \left\langle \underline{\mathsf{map}}_{\lambda\alpha\rho_1,\pi,\sigma}[f, x\mathsf{L}], \underline{\mathsf{map}}_{\lambda\alpha\rho_2,\pi,\sigma}[f, x\mathsf{R}] \right\rangle \\
\underline{\mathsf{map}}_{\lambda\alpha.\rho_1+\rho_2,\pi,\sigma} & = & x\mathrm{E}_+\left(\lambda y_1^{\rho_1[\pi]}.\mathsf{INL}_{\rho_2[\sigma]}\underline{\mathsf{map}}_{\lambda\alpha\rho_1,\pi,\sigma}[f, y_1]\right)\left(\lambda y_2^{\rho_2[\pi]}.\mathsf{INR}_{\rho_1[\sigma]}\underline{\mathsf{map}}_{\lambda\alpha\rho_2,\pi,\sigma}[f, y_2]\right) \\
\underline{\mathsf{map}}_{\lambda\alpha\rho,\pi,\sigma} & = & x\mathrm{E}_\exists\left(\Lambda\gamma\lambda y^{\tau[\pi]}.C_{\exists\gamma.\tau[\sigma],\gamma}\underline{\mathsf{map}}_{\lambda\alpha\tau,\pi,\sigma}[f, y]\right)
\end{array}
$$

It might now be instructive to reconsider the examples discussed in 3.5.

5.4 The term rewrite systems **MePIT** and **coMePIT**

These are systems which only serve a purpose in the circles of embeddings. Their main advantages over arbitrary NISPIT+ex and NIPIT, respectively, are their well-behaved map-terms not using sum types.
MePIT is defined as one of the systems ESMIT+ex. Let the underlying set MTypes be defined as the least set containing Typevars, 0 and 1 and being closed under $\to$, $\forall$, $\times$, $+$, $\exists$ and the following clause:

(μ) If $\rho \in \mathsf{MTypes}$ and $\alpha \notin \{\beta\} \cup \mathsf{FV}(\rho)$, then $\mu\alpha\exists\beta.(\beta \to \alpha) \times \rho \in \mathsf{MTypes}$.

It is very easy to see that MTypes is closed under substitution, i.e. the condition (sub) in the definition of ESMIT+ex is satisfied. It is also easy to see that MTypes is a subset of the type system of NISPIT+ex. Hence we define MePIT to be the restriction of NISPIT+ex to MTypes. The terms map do not use types outside MTypes. This is easily proved: The only case of interest is (μ). We see that for $\alpha \notin \{\beta\} \cup \mathsf{FV}(\rho)$ the definition yields

$$
\mathsf{map}_{\lambda\alpha\exists\beta.(\beta\to\alpha)\times\rho} = \Lambda\alpha\Lambda\gamma\lambda f^{\alpha\to\gamma}\lambda x^{\exists\beta.(\beta\to\alpha)\times\rho}.x\mathrm{E}_\exists(\Lambda\beta\lambda y^{(\beta\to\alpha)\times\rho}.C_{\exists\beta.(\beta\to\gamma)\times\rho,\beta}\langle\lambda z^\beta.f(y\mathsf{L}z), y\mathsf{R}\rangle),
$$

which is clearly okay.

The term rewrite system coMePIT is defined as one of the systems ESMIT. Let the underlying set MTypes be defined as the least set containing Typevars, 0 and 1 and being closed under $\to$, $\forall$, $\times$, $+$ and the following clause:

(μ) If $\rho \in \mathsf{MTypes}$ and $\alpha \notin \{\beta\} \cup \mathsf{FV}(\rho)$, then $\mu\alpha\forall\beta.(\alpha \to \beta) \to \rho \in \mathsf{MTypes}$.

It is again easy to see that MTypes is closed under substitution and that $\mathsf{MTypes} \subset \mathsf{NIPosTypes}$. Hence we define coMePIT to be the restriction of NIPIT to MTypes. Again it is easily proved that the terms map do not use types outside MTypes. Consider (μ). We see that for $\alpha \notin \{\beta\} \cup \mathsf{FV}(\rho)$ the definition gives us

$$
\mathsf{map}_{\lambda\alpha\forall\beta.(\alpha\to\beta)\to\rho} = \Lambda\alpha\Lambda\gamma\lambda f^{\alpha\to\gamma}\lambda x^{\forall\beta.(\alpha\to\beta)\to\rho}\Lambda\beta\lambda y^{\gamma\to\beta}.x\beta(\lambda z^\alpha.y(fz)),
$$

which indeed only uses types in MTypes.

Chapter 6

Term rewrite systems in the spirit of Mendler's system of inductive types

The following motivation became clear to me in quite late a stage of the work on this thesis although the report in [UV97] of an embedding of Mendler's system in [Men87a] into the system of positive inductive types suddenly clarified three mysterious facts on Mendler's system: That I succeeded in proving strong normalization without having to restrict to positive inductive types as Mendler did. (The elimination-based proof given in section 9.4.1 is practically my original proof of December 1996 presented at a workshop in Munich[1].) That iteration is faithfully represented by primitive recursion (which is now section 6.1.3) and that for positive inductive types without $+$ one can define a predecessor (which unrolls $\mu\alpha\rho$ to $\rho[\alpha:=\mu\alpha\rho]$) having the right reduction behaviour only after adding some η-rules which do no harm to confluence and strong normalization.
It is interesting that already in [Lei90, p. 308] the idea behind the elimination rule corresponding to iteration modelled after Mendler's recursion rule was identified (and called M-induction) but not used to get rid of the positivity restrictions.
Let $(U,\leq)$ be a complete lattice and $\Phi : U \to U$ any function. There are two canonical ways of producing a monotone function out of Φ. Define $\Phi^{\geq} : U \to U$ and $\Phi^{\leq} : U \to U$ by setting

$$\Phi^{\geq}(M) := \bigvee\{\Phi(M')|M' \leq M\}$$

and

$$\Phi^{\leq}(M) := \bigwedge\{\Phi(M')|M \leq M'\}.$$

$\Phi^{\geq}$ and $\Phi^{\leq}$ are monotone due to transitivity of $\leq$, and $\Phi^{\geq}(M) \geq \Phi(M)$ and $\Phi^{\leq}(M) \leq \Phi(M)$ due to reflexivity of $\leq$ (of course, some properties of suprema and infima enter the argument). Clearly, $\Phi^{\geq} = \Phi = \Phi^{\leq}$ for monotone Φ. And Φ is monotone, if $\Phi^{\geq} = \Phi$ or $\Phi^{\leq} = \Phi$.
Because $\Phi \mapsto \Phi^{\geq}$ and $\Phi \mapsto \Phi^{\leq}$ are also monotone, it is clear that $\Phi^{\geq}$ is the pointwise least monotone function being pointwise greater than Φ and that $\Phi^{\leq}$ is the pointwise greatest monotone function being pointwise smaller than Φ. Therefore I would like to call $\Phi^{\geq}$ the upper monotonization and $\Phi^{\leq}$ the lower monotonization of Φ.
By Tarski, there are least fixed points of $\Phi^{\geq}$ and $\Phi^{\leq}$. Let us study how their elimination and introduction rules may be expressed in terms of Φ.
Set $\mathsf{lfp}^{\geq}(\Phi) := \mathsf{lfp}(\Phi^{\geq})$. The rule ($\mathsf{lfp}$-E) says that $\Phi^{\geq}(M) \leq M \Rightarrow \mathsf{lfp}^{\geq}(\Phi) \leq M$. This is equivalent to

[1] Arbeitstagung Deduktive Aspekte von Beweistheorie und Informatik, Munich, December 19–20, 1996, organized by Gerhard Jäger (Bern) and Helmut Schwichtenberg (Munich).

($\mathsf{lfp}^{\geq}$-E) $(\forall M'.M' \leq M \Rightarrow \Phi(M') \leq M) \Rightarrow \mathsf{lfp}^{\geq}(\Phi) \leq M$.

The rule (lfp-E$^+$) says that $\Phi^{\geq}(\mathsf{lfp}^{\geq}(\Phi) \wedge M) \leq M \Rightarrow \mathsf{lfp}^{\geq}(\Phi) \leq M$. This is equivalent to

($\mathsf{lfp}^{\geq}$-E$^+$) $(\forall M'.M' \leq \mathsf{lfp}^{\geq}(\Phi) \Rightarrow M' \leq M \Rightarrow \Phi(M') \leq M) \Rightarrow \mathsf{lfp}^{\geq}(\Phi) \leq M$.

Set $\mathsf{lfp}^{\leq}(\Phi) := \mathsf{lfp}(\Phi^{\leq})$. The rule ($\mathsf{lfp}$-E) says that $\Phi^{\leq}(M) \leq M \Rightarrow \mathsf{lfp}^{\leq}(\Phi) \leq M$. Unfortunately, $\Phi^{\leq}(M) \leq M$ cannot be expressed in simpler terms in this general setting.
Let now $(U,\subseteq)$ be a complete lattice of sets. The rules ($\mathsf{lfp}^{\geq}$-E) and ($\mathsf{lfp}^{\geq}$-E$^+$) may be expressed as

($\mathsf{lfp}^{\supseteq}$-E) If $x \in \mathsf{lfp}^{\supseteq}(\Phi)$ and $\forall M'.M' \subseteq M \Rightarrow \Phi(M') \subseteq M$, then $x \in M$.

($\mathsf{lfp}^{\supseteq}$-E$^+$) If $x \in \mathsf{lfp}^{\supseteq}(\Phi)$ and $\forall M'.M' \subseteq \mathsf{lfp}^{\supseteq}(\Phi) \Rightarrow M' \subseteq M \Rightarrow \Phi(M') \subseteq M$, then $x \in M$.

This motivates the elimination rules of MeIT and UVIT whose common type system imposes no restrictions on the formation on $\mu\alpha\rho$ reflecting the use of any Φ:

(μ-E) If $r^{\mu\alpha\rho} \in \Lambda$ and $s^{\forall\alpha.(\alpha\to\sigma)\to\rho\to\sigma} \in \Lambda$ (with $\alpha \notin \mathsf{FV}(\sigma)$), then $(r\mathrm{E}_\mu s)^\sigma \in \Lambda$.

(μ-E$^+$) If $r^{\mu\alpha\rho} \in \Lambda$ and $s^{\forall\alpha.(\alpha\to\mu\alpha\rho)\to(\alpha\to\sigma)\to\rho\to\sigma} \in \Lambda$ (with $\alpha \notin \mathsf{FV}(\sigma)$), then $(r\mathrm{E}^+_\mu s)^\sigma \in \Lambda$.

MeIT and UVIT will differ in the introduction rule. Let us go back to the general situation of $(U,\leq)$: The fact that $\mathsf{lfp}^{\geq}(\Phi)$ is a pre-fixed-point of $\Phi^{\geq}$ is equivalent to

($\mathsf{lfp}^{\geq}$-I) $\forall M.M \leq \mathsf{lfp}^{\geq}(\Phi) \Rightarrow \Phi(M) \leq \mathsf{lfp}^{\geq}(\Phi)$.

As a trivial (due to reflexivity of $\leq$) instance we get

($\mathsf{lfp}^{\geq}$-I-wk) $\Phi(\mathsf{lfp}^{\geq}(\Phi)) \leq \mathsf{lfp}^{\geq}(\Phi)$.

For $(U,\subseteq)$ a complete lattice of sets we get

($\mathsf{lfp}^{\supseteq}$-I) If $M \subseteq \mathsf{lfp}^{\geq}(\Phi)$ and $x \in \Phi(M)$, then $x \in \mathsf{lfp}^{\supseteq}(\Phi)$.

($\mathsf{lfp}^{\supseteq}$-I-wk) If $x \in \Phi(\mathsf{lfp}^{\supseteq}(\Phi))$, then $x \in \mathsf{lfp}^{\supseteq}(\Phi)$.

This directly motivates the introduction rules for UVIT and MeIT. For UVIT we take

(μ-I) If $j^{\tau\to\mu\alpha\rho} \in \Lambda$ and $t^{\rho[\alpha:=\tau]} \in \Lambda$, then $(Cjt)^{\mu\alpha\rho} \in \Lambda$ and $\mathsf{FV}(Cjt) := \mathsf{FV}(j) \cup \mathsf{FV}(t)$.

corresponding to ($\mathsf{lfp}^{\supseteq}$-I) and for MeIT the introduction rule

(μ-I) If $t^{\rho[\alpha:=\mu\alpha\rho]} \in \Lambda$, then $(C_{\mu\alpha\rho}t)^{\mu\alpha\rho} \in \Lambda$.

corresponding to ($\mathsf{lfp}^{\supseteq}$-I-wk).
The reader should be warned that the introduction rule for MeIT seems to be coming from (lfp-I) depending on a monotone Φ. Although the rule pertaining to (lfp-I) is in fact the same (e. g. in IMIT), an understanding of MeIT would be impossible[2].
We did not yet model $\mathsf{lfp}^{\leq}(\Phi)$ in a term system. Let now $(U,\subseteq)$ be a complete lattice of sets where the infimum is always given by set intersection. Hence,

$$\Phi^{\subseteq}(M) = \{x|\forall M'.M \subseteq M' \Rightarrow x \in \Phi(M')\}.$$

The rules (lfp-E) and (lfp-E$^+$) therefore give for $\Phi^{\subseteq}$ instead of Φ the following rules

[2] Presumably it is this wrong intuition which gave Mendler's system a flavour of "mystery" (see the introductory remarks to this section).

($\mathsf{lfp}^{\subseteq}$-E) If $x \in \mathsf{lfp}^{\subseteq}(\Phi)$ and $\forall y.\Big(\forall M'.M \subseteq M' \Rightarrow y \in \Phi(M')\Big) \Rightarrow y \in M$, then $x \in M$.

($\mathsf{lfp}^{\subseteq}$-E$^+$) If $x \in \mathsf{lfp}^{\subseteq}(\Phi)$ and $\forall y.\Big(\forall M'.\mathsf{lfp}^{\subseteq}(\Phi) \wedge M \subseteq M' \Rightarrow y \in \Phi(M')\Big) \Rightarrow y \in M$, then $x \in M$.

They serve as the motivation for the elimination rules of coMeIT (also without restrictions to forming $\mu\alpha\rho$):

(μ-E) If $r^{\mu\alpha\rho} \in \Lambda$ and $s^{(\forall\alpha.(\sigma\to\alpha)\to\rho)\to\sigma} \in \Lambda$ (with $\alpha \notin \mathsf{FV}(\sigma)$), then $(r\mathrm{E}_\mu s)^\sigma \in \Lambda$.

(μ-E$^+$) If $r^{\mu\alpha\rho} \in \Lambda$ and $s^{(\forall\alpha.(\mu\alpha\rho\times\sigma\to\alpha)\to\rho)\to\sigma} \in \Lambda$ (with $\alpha \notin \mathsf{FV}(\sigma)$), then $(r\mathrm{E}^+_\mu s)^\sigma \in \Lambda$.

The fact that $\mathsf{lfp}^{\subseteq}(\Phi)$ is a pre-fixed-point of $\Phi^{\subseteq}$ is equivalent to the following rule

($\mathsf{lfp}^{\subseteq}$-I) If $\forall M.\mathsf{lfp}^{\subseteq}(\Phi) \subseteq M \Rightarrow x \in \Phi(M)$, then $x \in \mathsf{lfp}^{\subseteq}(\Phi)$.

This directly motivates the introduction rule of coMeIT:

(μ-I) If $t^{\forall\alpha.(\mu\alpha\rho\to\alpha)\to\rho} \in \Lambda$ then $(C_{\mu\alpha\rho}t)^{\mu\alpha\rho} \in \Lambda$.

What is the purpose of this chapter? The systems MeIT, UVIT and coMeIT are introduced to elucidate the embeddings of the systems of monotone inductive types into systems of non-interleaved positive (or—in the presence of the existential quantifier—even strictly positive) inductive types.
The idea of forming $\Phi^{\supseteq}$ and $\Phi^{\subseteq}$ will later also be used in the direct proofs of strong normalization of the systems of monotone inductive types, and hence it is interesting to isolate these concepts in some formal system.
The embeddings given in this chapter all draw from the lattice-theoretic intuition: The embeddings of the newly introduced systems into systems of positive inductive types are derived from analyzing the above arguments justifiying the rules, whereas the embeddings of the systems of monotone inductive types into MeIT et al come from the fact that for monotone Φ (whence $\Phi^{\supseteq} = \Phi = \Phi^{\subseteq}$) the original introduction and elimination rules for least fixed points may be derived from those for $\mathsf{lfp}^{\supseteq}$ and $\mathsf{lfp}^{\subseteq}$, respectively. The latter fact is not at all surprising but the impact it has due to the compatibility of its proof with reduction: Monotone inductive types embed into non-interleaving positive inductive types.
In order to reduce notational overhead in the direct proofs of strong normalization for MeIT and coMeIT, it is shown that iteration may be easily derived from recursion in these systems (contrast this with the situation of monotone inductive types discussed in section 4.5).
The picture of the systems is completed by embedding their iterative fragments into system F.

6.1 The variant **MeIT** of a term rewrite system proposed by Nax Mendler

The system corresponding to ($\mathsf{lfp}^{\supseteq}$-I-wk), ($\mathsf{lfp}^{\supseteq}$-E) and ($\mathsf{lfp}^{\supseteq}$-E$^+$) is defined and its connection with Mendler's system in [Men87a] discussed. The system varIMIT is embedded into it and it is embedded into its fragment without iteration and its iterative fragment embedded into eF.

6.1.1 Definition

The definition is again quite elaborate because a direct normalization proof (for MeIT−it) will be given in section 9.4.1.
The (second) definition of typed terms and their free variables for system eF is extended by the following clauses:

(μ-I) If $t^{\rho[\alpha:=\mu\alpha\rho]} \in \Lambda$, then $(C_{\mu\alpha\rho}t)^{\mu\alpha\rho} \in \Lambda$ and $\mathsf{FV}(C_{\mu\alpha\rho}t) := \mathsf{FV}(t)$.

(μ-E) If $r^{\mu\alpha\rho} \in \Lambda$ and $s^{\forall\alpha.(\alpha\to\sigma)\to\rho\to\sigma} \in \Lambda$ (with $\alpha \notin \mathsf{FV}(\sigma)$), then $(rs)^\sigma \in \Lambda$ (also written as $r\mathrm{E}_\mu\sigma s$ or shorter $r\mathrm{E}_\mu s$) and $\mathsf{FV}(rs) := \mathsf{FV}(r) \cup \mathsf{FV}(s)$.

(μ-E$^+$) If $r^{\mu\alpha\rho} \in \Lambda$ and $s^{\forall\alpha.(\alpha\to\mu\alpha\rho)\to(\alpha\to\sigma)\to\rho\to\sigma} \in \Lambda$ (with $\alpha \notin \mathsf{FV}(\sigma)$), then $(rs)^\sigma \in \Lambda$ (also written as $r\mathrm{E}^+_\mu\sigma s$ or shorter $r\mathrm{E}^+_\mu s$) and $\mathsf{FV}(rs) := \mathsf{FV}(r) \cup \mathsf{FV}(s)$.

This definition is again compatible with the renaming convention for bound type variables.
Extend the recursive definition of $\mathsf{FTV}(r)$ by the following clauses:

(μ-I) $\mathsf{FTV}(C_{\mu\alpha\rho}t) := \mathsf{FV}(\mu\alpha\rho) \cup \mathsf{FTV}(t)$.

(μ-E) $\mathsf{FTV}(rs) := \mathsf{FTV}(r) \cup \mathsf{FTV}(s)$.

(μ-E$^+$) $\mathsf{FTV}(rs) := \mathsf{FTV}(r) \cup \mathsf{FTV}(s)$.

The definition of $r[\vec{x}^{\vec{\rho}} := \vec{s}]$ together with the proof of $r^\rho[\vec{x}^{\vec{\rho}} := \vec{s}] : \rho$ and $\mathsf{FV}(r[\vec{x}^{\vec{\rho}} := \vec{s}]) = (\mathsf{FV}(r) \setminus \vec{x}^{\vec{\rho}}) \cup \bigcup\{\mathsf{FV}(s_i) | x_i^{\rho_i} \in \mathsf{FV}(r) \wedge x_i^{\rho_i} \neq x_j^{\rho_j} \text{ for } j < i\}$ (i. e., the statement of Lemma 2.10) is extended by the following clauses:

(μ-I) $(C_{\mu\alpha\rho}t)[\vec{x}^{\vec{\rho}} := \vec{s}] := C_{\mu\alpha\rho}t[\vec{x}^{\vec{\rho}} := \vec{s}]$. Proof obivous.

(μ-E) $(rs)[\vec{x}^{\vec{\rho}} := \vec{s}] := r[\vec{x}^{\vec{\rho}} := \vec{s}]\mathrm{E}_\mu s[\vec{x}^{\vec{\rho}} := \vec{s}]$. Proof obvious.

(μ-E$^+$) $(rs)[\vec{x}^{\vec{\rho}} := \vec{s}] := r[\vec{x}^{\vec{\rho}} := \vec{s}]\mathrm{E}^+_\mu s[\vec{x}^{\vec{\rho}} := \vec{s}]$. Proof obvious.

The definition of $r[\vec{\alpha} := \vec{\sigma}]$ together with the proof of $r^\rho[\vec{\alpha} := \vec{\sigma}] : \rho[\vec{\alpha} := \vec{\sigma}]$ and $\mathsf{FV}(r[\vec{\alpha} := \vec{\sigma}]) = \{y^{\tau[\vec{\alpha}:=\vec{\sigma}]} | y^\tau \in \mathsf{FV}(r)\}$ is extended by the following clauses:

(μ-I) $(C_{\mu\alpha\rho}t)[\vec{\alpha} := \vec{\sigma}] := C_{(\mu\alpha\rho)[\vec{\alpha}:=\vec{\sigma}]}t[\vec{\alpha} := \vec{\sigma}]$. Proof obvious.

(μ-E) $(rs)[\vec{\alpha} := \vec{\sigma}] := r[\vec{\alpha} := \vec{\sigma}]\mathrm{E}_\mu s[\vec{\alpha} := \vec{\sigma}]$. Proof obvious.

(μ-E$^+$) $(rs)[\vec{\alpha} := \vec{\sigma}] := r[\vec{\alpha} := \vec{\sigma}]\mathrm{E}^+_\mu s[\vec{\alpha} := \vec{\sigma}]$. Proof obvious.

The statements made in section 2.1.2 are valid in MeIT, too.
The general definition of immediate subterms of a term gives the following extension for system MeIT: $C_{\mu\alpha\rho}t$ has exactly the immediate subterms t and $r\mathrm{E}_\mu s$ and $r\mathrm{E}^+_\mu s$ have exactly the immediate subterms r and s.
Extend the inductive definition of $r\vec{s}$ for system eF as follows:

(μ-E) If $r\vec{s} : \mu\alpha\rho$ and $s : \forall\alpha.(\alpha \to \sigma) \to \rho \to \sigma$ (with $\alpha \notin \mathsf{FV}(\sigma)$), then $r(\vec{s}, s) := (r\vec{s})s$.

(μ-E$^+$) If $r\vec{s} : \mu\alpha\rho$ and $s : \forall\alpha.(\alpha \to \mu\alpha\rho) \to (\alpha \to \sigma) \to \rho \to \sigma$ (with $\alpha \notin \mathsf{FV}(\sigma)$), then $r(\vec{s}, s) := (r\vec{s})s$.

The statement of Lemma 2.23 holds also true for MeIT.

Lemma 6.1 (Head form) Every term has either exactly one of the forms given in Lemma 2.44 or exactly one of the following forms:

(μ-I) $C_{\mu\alpha\rho}t$

(μ-R) $(C_{\mu\alpha\rho}t)s\vec{s}$, subsuming both eliminations for μ

Proof Induction on terms. □

Extend the inductive definition of the set NF for system eF by the following clause:

(μ-I) If $C_{\mu\alpha\rho}t$ is a term and $t \in \mathsf{NF}$, then $C_{\mu\alpha\rho}t \in \mathsf{NF}$.

The statements made in section 2.1.3 are all valid in MeIT.
Define the relation $\mapsto$ as for system eF, but add the following clauses:

MeIT

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

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

In both rules we require that $x^{\mu\alpha\rho} \notin \mathsf{FV}(s)$.
As for system eF the objects on the right side are automatically terms if the respective object on the left is a term, and the types are equal.
The relation $\to_{whd}$ is defined as for system F and has the same properties as stated there.
Extend the inductive definition of the relation $\to$ for system eF by the congruence rules for μ and prove that if $r \to r'$, then r and r' have the same type and $\mathsf{FV}(r') \subseteq \mathsf{FV}(r)$.
Extend the inductive definition of WN for system eF by the following clauses:

(μ-I) If $C_{\mu\alpha\rho}t$ is a term and $t \in \mathsf{WN}$, then $C_{\mu\alpha\rho}t \in \mathsf{WN}$.

(β_μ) If $s(\mu\alpha\rho)(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma s)t\vec{s} \in \mathsf{WN}$, $x^{\mu\alpha\rho} \notin \mathsf{FV}(s)$ and $(C_{\mu\alpha\rho}t)\mathrm{E}_\mu\sigma s\vec{s}$ is a term, then $(C_{\mu\alpha\rho}t)\mathrm{E}_\mu\sigma s\vec{s} \in \mathsf{WN}$.

(β_μ^+) If $s(\mu\alpha\rho)(\lambda y^{\mu\alpha\rho}y)(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma s)t\vec{s} \in \mathsf{WN}$, $x^{\mu\alpha\rho} \notin \mathsf{FV}(s)$ and $(C_{\mu\alpha\rho}t)\mathrm{E}_\mu^+\sigma s\vec{s}$ is a term, then $(C_{\mu\alpha\rho}t)\mathrm{E}_\mu^+\sigma s\vec{s} \in \mathsf{WN}$.

Again definition by recursion on WN is admissible. Extend the recursive definition of the function Ω from WN to Λ and the simultaneous proof of $r \to^* \Omega(r) \in \mathsf{NF}$ for $r \in \mathsf{WN}$ by the following clauses:

(μ-I) $\Omega(C_{\mu\alpha\rho}t) := C_{\mu\alpha\rho}\Omega(t)$.

(β_μ) $\Omega((C_{\mu\alpha\rho}t)\mathrm{E}_\mu\sigma s\vec{s}) := \Omega\Big(s(\mu\alpha\rho)(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma s)t\vec{s}\Big)$.

(β_μ^+) $\Omega((C_{\mu\alpha\rho}m)\mathrm{E}_\mu^+\sigma s\vec{s}) := \Omega\Big(s(\mu\alpha\rho)(\lambda y^{\mu\alpha\rho}y)(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma s)t\vec{s}\Big)$.

The proofs are always obvious.
Extend the definition of SN for system eF by the following clauses:

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

(β_μ) If $s(\mu\alpha\rho)(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma s)t\vec{s} \in \mathsf{SN}$, $x^{\mu\alpha\rho} \notin \mathsf{FV}(s)$ and $(C_{\mu\alpha\rho}t)\mathrm{E}_\mu\sigma s\vec{s}$ is a term, then $(C_{\mu\alpha\rho}t)\mathrm{E}_\mu\sigma s\vec{s} \in \mathsf{SN}$.

(β_μ^+) If $s(\mu\alpha\rho)(\lambda y^{\mu\alpha\rho}y)(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma s)t\vec{s} \in \mathsf{SN}$, $x^{\mu\alpha\rho} \notin \mathsf{FV}(s)$ and $(C_{\mu\alpha\rho}t)\mathrm{E}_\mu^+\sigma s\vec{s}$ is a term, then $(C_{\mu\alpha\rho}t)\mathrm{E}_\mu^+\sigma s\vec{s} \in \mathsf{SN}$.

The only difference between the μ-clauses for SN and WN is the name of the defined set.
Every statement made in section 2.1.4 for system F (on nf, NF, wn, WN, Ω, SN and sn) is also true for MeIT.
MeIT is not the original system of [Men87a]: The minor differences are as follows:

- In MeIT we have no coinductive types which however could be treated analogously.
- Iteration has been added although it may be eliminated as shown in section 6.1.3.
- Recursion is not expressed via a recursion operator but by a rule.
- MeIT is defined as an extension of eF and not only of F.

In contrast to the original system we allow any $\mu\alpha\rho$ without positivity requirement and have the introduction and elimination rules for any $\mu\alpha\rho$. This will be essential for the embedding of varIMIT in MeIT in the next section.

6.1.2 Embedding varIMIT in MeIT

In the introduction to this chapter it was shown that ($\mathsf{lfp}^{\supseteq}$-E) and ($\mathsf{lfp}^{\supseteq}$-E$^+$) are equivalent to (lfp-E) and (lfp-E$^+$) if one replaced Φ by $\Phi^{\supseteq}$ in the latter rules. The rule ($\mathsf{lfp}^{\supseteq}$-I-wk) was taken as a consequence of (lfp-I) where again Φ was replaced by $\Phi^{\supseteq}$. If Φ is monotone, then $\Phi^{\supseteq} = \Phi$ and hence the replacement does not change anything[3]. Therefore, in this case (lfp-E) and (lfp-E$^+$) are consequences of ($\mathsf{lfp}^{\supseteq}$-E) and ($\mathsf{lfp}^{\supseteq}$-E$^+$), respectively. Moreover, although ($\mathsf{lfp}^{\supseteq}$-I-wk) only is an instance of (lfp-I) with Φ replaced by $\Phi^{\supseteq}$, it is the same rule for $\Phi^{\supseteq} = \Phi$. Because of the latter fact we may take the homomorphic μ-introduction rule in the following definition. The clauses pertaining to elimination reflect the proof of the other consequences—more precisely: They reflect the proofs that for monotone Φ (lfp-E) and (lfp-E$^+$) are consequences of ($\mathsf{lfp}^{\supseteq}$-E) and ($\mathsf{lfp}^{\supseteq}$-E$^+$) where $\mathsf{lfp}^{\supseteq}$ has been replaced by lfp in the latter rules. And because these proofs only need monotonicity in a restricted way an embedding even for varIMIT is gained and not only for IMIT[4].
Set $\rho' := \rho$ (for ρ a type in varIMIT) and for every term r^ρ of system varIMIT define the term r' of system MeIT by recursion on r and simultaneously prove that $r' : \rho$ and $\mathsf{FTV}(r') = \mathsf{FTV}(r)$ and $\mathsf{FV}(r') = \mathsf{FV}(r)$ as follows:

(eF) The homomorphic term rules for eF.

(μ-I) The homomorphic μ-introduction rule.

(μ-E) $(r^{\mu\alpha\rho}\mathrm{E}_\mu\sigma ms)' := r'\mathrm{E}_\mu\sigma\hat{s}$ with $\hat{s} := \Lambda\alpha\lambda z^{\alpha\to\sigma}\lambda y^\rho.s'(m'\alpha zy)$, where we assume that $\alpha \notin \mathsf{FTV}(s) \cup \mathsf{FTV}(m)$ and require $z, y \notin \mathsf{FV}(s) \cup \mathsf{FV}(m)$ and $z \neq y$.

(μ-E$^+$) $(r^{\mu\alpha\rho}\mathrm{E}_\mu^+\sigma ms)' := r'\mathrm{E}_\mu^+\sigma\hat{s}$ with $\hat{s} := \Lambda\alpha\lambda z_1^{\alpha\to\mu\alpha\rho}\lambda z_2^{\alpha\to\sigma}\lambda y^\rho.s'(m'\alpha(\lambda x^\alpha\langle z_1x, z_2x\rangle)y)$, where we assume that $\alpha \notin \mathsf{FTV}(s) \cup \mathsf{FTV}(m)$ and require $z_1, z_2, y \notin \mathsf{FV}(s) \cup \mathsf{FV}(m)$ and z_1, z_2, y different.

[3] This explains why we take $\rho' := \rho$ in the embedding. It is the clue to the problem of embedding term systems with monotonicity witnesses into term systems with positivity restrictions in the types which therefore guarantee monotonicity even on the level of types.

[4] An embedding of EMIT into MeIT cannot be constructed in this easy way because monotonicity enters the arguments for the elimination rules where EMIT does not have the monotonicity witnesses.

(The proofs are obvious.) Recall that $\alpha \notin \mathsf{FV}(\sigma)$ is part of the definition of the μ-rules of varIMIT, which implies that α is not free in the type of s (in both rules). Therefore the assumption $\alpha \notin \mathsf{FTV}(s)$ is admissible.
The statement of Lemma 4.16 is true for this definition and is proved by induction on r. That $-'$ is an embedding is proved by induction on $\to$. Only (β_μ) and (β_μ^+) deserve attention. Check e. g.

$$
\begin{array}{lrcl}
(\beta_\mu^+): & ((C_{\mu\alpha\rho}t)\mathrm{E}_\mu^+\sigma ms)' & = & (C_{\mu\alpha\rho}t')\mathrm{E}_\mu^+\sigma\hat{s} \quad \text{with } \hat{s} \text{ as defined above} \\
& & \mapsto & \hat{s}(\mu\alpha\rho)(\lambda y^{\mu\alpha\rho}y)(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma\hat{s})t' \\
& & \to^4 & s'\Big(m'(\mu\alpha\rho)\Big(\lambda x^{\mu\alpha\rho}.\langle(\lambda y^{\mu\alpha\rho}y)x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma\hat{s})x\rangle\Big)t'\Big) \\
& & \to & s'\Big(m'(\mu\alpha\rho)\Big(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma\hat{s})x\rangle\Big)t'\Big) \\
& & = & \Big(s\Big(m(\mu\alpha\rho)\Big(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma ms)x\rangle\Big)t\Big)\Big)'.
\end{array}
$$

6.1.3 Embedding MeIT in MeIT–it

The following embedding is nearly trivial. Mendler's original system in [Men87a] does not have an iterator. This section shows that an iterator—in MeIT the rule (μ-E)—is superfluous. Note that this reduction is not directly possible for the systems of monotone inductive types (see section 4.5).
Let MeIT–it be the system which is derived from MeIT by leaving out the term rule (μ-E) and the reduction rule (β_μ).
Set $\rho' := \rho$ and for every term r^ρ of system MeIT define the term r' of system MeIT–it by recursion on r and simultaneously prove that $r' : \rho$ and $\mathsf{FTV}(r') = \mathsf{FTV}(r)$ and $\mathsf{FV}(r') = \mathsf{FV}(r)$ as follows:

(eF) The homomorphic term rules for eF.

(μ-I) The homomorphic μ-introduction rule.

(μ-E) $(r^{\mu\alpha\rho}\mathrm{E}_\mu\sigma s)' := r'\mathrm{E}_\mu^+\sigma\hat{s}$ with $\hat{s} := \Lambda\alpha\lambda z^{\alpha\to\mu\alpha\rho}.s'\alpha$, where we assume that $\alpha \notin \mathsf{FTV}(s)$ and require $z \notin \mathsf{FV}(s)$.

(μ-E$^+$) The homomorphic μ^+-elimination rule.

(The proofs are obvious.) Substitutivity (the statement of Lemma 4.16) is true for this definition and is proved by induction on r. That $-'$ is an embedding is proved by induction on $\to$. Only the rule (β_μ) deserves attention.

$$
\begin{array}{lrcl}
(\beta_\mu): & ((C_{\mu\alpha\rho}t)\mathrm{E}_\mu\sigma s)' & = & (C_{\mu\alpha\rho}t')\mathrm{E}_\mu^+\sigma\hat{s} \quad \text{with } \hat{s} \text{ as defined above} \\
& & \mapsto & \hat{s}(\mu\alpha\rho)(\lambda y^{\mu\alpha\rho}y)(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma\hat{s})t' \\
& & \to^2 & s'(\mu\alpha\rho)(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma\hat{s})t' \\
& & = & \Big(s(\mu\alpha\rho)(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma s)t\Big)'.
\end{array}
$$

6.1.4 Embedding MeIT–rec in eF

Because I have no idea for a direct embedding of IMIT–rec into IT, MeIT–rec is embedded into eF. (Via the circle of embeddings we finally get that IMIT–rec embeds into eF.)
Let MeIT–rec be the system which is derived from MeIT by leaving out the term generation rule (μ-E$^+$) and the reduction rule (β_μ^+).
For every type ρ of system MeIT–rec define the type ρ' of system eF by recursion on ρ and simultaneously prove that $\mathsf{FV}(\rho') = \mathsf{FV}(\rho)$ as follows:

(eF) The homomorphic type rules for eF.

(μ) $(\mu\alpha\rho)' := \forall\beta.(\forall\alpha.(\alpha\to\beta)\to\rho'\to\beta)\to\beta$ for $\beta \notin \{\alpha\} \cup \mathsf{FV}(\rho)$.

(The proofs were obvious.) This definition is compatible with the variable conventions for the systems MeIT–rec and eF.

Lemma 6.2 $(\rho[\vec{\alpha} := \vec{\sigma}])' = \rho'[\vec{\alpha} := \vec{\sigma}']$.

Proof Induction on ρ. □

For every term r^ρ of system MeIT–rec define the term r' of system eF by recursion on r and simultaneously prove that $r' : \rho'$ and $\mathsf{FTV}(r') = \mathsf{FTV}(r)$ and $\mathsf{FV}(r') = \{x^{\sigma'} | x^\sigma \in \mathsf{FV}(r)\}$ as follows:

(eF) The homomorphic term rules for eF.

(μ-I) $(C_{\mu\alpha\rho}t)' := \Lambda\beta\lambda z^{\forall\alpha.(\alpha\to\beta)\to\rho'\to\beta}.z(\mu\alpha\rho)'(\lambda x^{(\mu\alpha\rho)'}.x\beta z)t'$ for $\beta \notin \{\alpha\} \cup \mathsf{FV}(\rho) \cup \mathsf{FTV}(t)$ and $z \notin \mathsf{FV}(t)$.

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

(The proofs were always obvious.) This definition is again compatible with the variable conventions for MeIT–rec and eF.
The statement of Lemma 4.11 is true for $-'$ and proved by induction on the terms. That $-'$ is an embedding is proved by induction on $\to$. Only (β_μ) deserves our attention:

$$
\begin{array}{rcl}
((C_{\mu\alpha\rho}t)\mathrm{E}_\mu\sigma s)' & = & (\Lambda\beta\lambda z^{\forall\alpha.(\alpha\to\beta)\to\rho'\to\beta}.z(\mu\alpha\rho)'(\lambda x^{(\mu\alpha\rho)'}.x\beta z)t')\sigma's' \\
& \to & (\lambda z^{\forall\alpha.(\alpha\to\sigma')\to\rho'\to\sigma'}.z(\mu\alpha\rho)'(\lambda x^{(\mu\alpha\rho)'}.x\sigma'z)t')s' \\
& \mapsto & s'(\mu\alpha\rho)'(\lambda x^{(\mu\alpha\rho)'}.x\sigma's')t' \\
& = & \Big(s(\mu\alpha\rho)(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma s)t\Big)'.
\end{array}
$$

6.2 The term rewrite system UVIT

The following system UVIT is a variant of MeIT which corresponds to the logical system $\mathrm{M}\mathbf{NI}\mathrm{P}_{\mathbf{r}}^{\mu,\nu}$ in [UV97]. The name UVIT (system of inductive types à la Uustalu and Vene) intends to honour [UV97] because it is the true junction between IMIT and NISPIT+ex, MeIT being only some peculiar variant of UVIT on the way from IMIT to NISPIT+ex as is the variant varIMIT of IMIT.

6.2.1 Definition

In what follows only changes to MeIT are given. All the statements on MeIT hold as well in UVIT. Changed clause in the definition of terms:

(μ-I) If $j^{\tau\to\mu\alpha\rho} \in \Lambda$ and $t^{\rho[\alpha:=\tau]} \in \Lambda$, then $(Cjt)^{\mu\alpha\rho} \in \Lambda$ and $\mathsf{FV}(Cjt) := \mathsf{FV}(j) \cup \mathsf{FV}(t)$.

Changed clauses in the definition of reduction:

UVIT		
	(β_μ)	$(Cj^{\tau\to\mu\alpha\rho}t)\mathrm{E}_\mu\sigma s \mapsto s\tau(\lambda x^\tau.(jx)\mathrm{E}_\mu\sigma s)t$
	(β_μ^+)	$(Cj^{\tau\to\mu\alpha\rho}t)\mathrm{E}_\mu^+\sigma s \mapsto s\tau(\lambda x^\tau.jx)(\lambda x^\tau.(jx)\mathrm{E}_\mu^+\sigma s)t$

In both cases we require that $x^\tau \notin \mathsf{FV}(j) \cup \mathsf{FV}(s)$.
All the other definitions may be adapted very easily from MeIT.

6.2.2 Embedding **MeIT** in **UVIT**

The following trivial embedding reflects the fact that $(\mathsf{lfp}^{\supseteq}\text{-I-wk})$ is weaker than $(\mathsf{lfp}^{\supseteq}\text{-I})$.
Set $\rho' := \rho$ for every type ρ of MeIT.
Define for every term r of system MeIT the term r' of UVIT by recursion on r and simultaneously prove that for $r : \rho$ we have $r' : \rho$ and $\mathsf{FTV}(r') = \mathsf{FTV}(r)$ and $\mathsf{FV}(r') = \mathsf{FV}(r)$ as follows:

(eF) The homomorphic term rules for eF.

(μ-I) $(C_{\mu\alpha\rho}t)' := C(\lambda y^{\mu\alpha\rho}y)t'$.

(μ-E$^{(+)}$) The homomorphic μ-elimination rules.

(The proofs are trivial.) The statement of Lemma 4.16 is true for this definition and is proved by induction on r. That $-'$ is an embedding is proved by induction on $\to$. Only (β_μ) and (β_μ^+) deserve attention. Check e. g.

$$\begin{array}{lrcl}(\beta_\mu^+): & ((C_{\mu\alpha\rho}t)\mathrm{E}_\mu^+\sigma s)' & = & (C(\lambda y^{\mu\alpha\rho}y)t')\mathrm{E}_\mu^+\sigma s' \\ & & \mapsto & s'(\mu\alpha\rho)(\lambda x^{\mu\alpha\rho}.(\lambda y^{\mu\alpha\rho}y)x)(\lambda x^{\mu\alpha\rho}.(\lambda y^{\mu\alpha\rho}y)x\mathrm{E}_\mu^+\sigma s')t' \\ & & \to^2 & s'(\mu\alpha\rho)(\lambda y^{\mu\alpha\rho}y)(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma s')t' \\ & & = & \Big(s(\mu\alpha\rho)(\lambda y^{\mu\alpha\rho}y)(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma s)t\Big)'.\end{array}$$

6.2.3 Embedding **UVIT** in **MePIT**

This section is nothing but the syntactic analogue of the considerations which motivated the definition of UVIT including the verification that reduction is preserved.
For every type ρ of system UVIT define the type ρ' of system MePIT by recursion on ρ and simultaneously prove that $\mathsf{FV}(\rho') = \mathsf{FV}(\rho)$ as follows:

(eF) The homomorphic type rules for eF.

(μ) $(\mu\alpha\rho)' := \mathsf{spos}(\mu\alpha\rho') = \mu\alpha\exists\beta.(\beta \to \alpha) \times \rho'[\alpha := \beta]$ for $\beta \notin \{\alpha\} \cup \mathsf{FV}(\rho)$ which is a type of MePIT because $\alpha \notin \{\beta\} \cup \mathsf{FV}(\rho'[\alpha := \beta])$.

(The proofs were obvious.) This definition is compatible with the variable conventions for the systems UVIT and MePIT.

Lemma 6.3 $(\rho[\vec{\alpha} := \vec{\sigma}])' = \rho'[\vec{\alpha} := \vec{\sigma}']$.

Proof Induction on ρ. □

Define for every term r of system UVIT the term r' of MePIT by recursion on r and simultaneously prove that $r' : \rho'$ and $\mathsf{FTV}(r') = \mathsf{FTV}(r)$ and $\mathsf{FV}(r') = \{x^{\sigma'} | x^\sigma \in \mathsf{FV}(r)\}$ as follows:

(eF) The homomorphic term rules for eF.

(μ-I) $(Cj^{\tau\to\mu\alpha\rho}t^{\rho[\alpha:=\tau]})' := C_{(\mu\alpha\rho)'}\hat{t}$ with $\hat{t} := C_{\exists\beta.(\beta\to(\mu\alpha\rho)')\times\rho'[\alpha:=\beta],\tau'}\langle j', t'\rangle$

(μ-E) $(r^{\mu\alpha\rho}\mathrm{E}_\mu\sigma s^{\forall\alpha.(\alpha\to\sigma)\to\rho\to\sigma})' := r'\mathrm{E}_\mu\sigma'\hat{s}$ with

$$\hat{s} := \lambda z^{\exists\alpha.(\alpha\to\sigma')\times\rho'}.z(\Lambda\alpha\lambda y^{(\alpha\to\sigma')\times\rho'}.s'\alpha(y\mathsf{L})(y\mathsf{R}))$$

(μ-E$^+$) $(r^{\mu\alpha\rho}\mathrm{E}_\mu^+\sigma s^{\forall\alpha.(\alpha\to\mu\alpha\rho)\to(\alpha\to\sigma)\to\rho\to\sigma})' := r'\mathrm{E}_\mu^+\sigma'\hat{s}$ with

$$\hat{s} := \lambda z^{\exists\alpha.(\alpha\to(\mu\alpha\rho)'\times\sigma')\times\rho'}.z(\Lambda\alpha\lambda y^{(\alpha\to(\mu\alpha\rho)'\times\sigma')\times\rho'}.s'\alpha(\lambda x^\alpha.y\mathsf{L}x\mathsf{L})(\lambda x^\alpha.y\mathsf{L}x\mathsf{R})(y\mathsf{R}))$$

(The proofs are trivial.) The passage from s' to $\hat{s}$ in both elimination cases is nothing but uncurrying.
Substitutivity (the statement of Lemma 4.11) is again true for this definition.
By induction on $\to$ show that $-'$ indeed is an embedding. The only cases of interest are (β_μ) and (β_μ^+). They require a long calculation which essentially only shows that uncurrying is well-behaved with respect to reduction. We only check (β_μ^+): $((Cjt)\mathrm{E}_\mu^+\sigma s)' \mapsto_{\beta_\mu^+}$
$\hat{s}\Big(\mathsf{map}_{\lambda\alpha\exists\beta.(\beta\to\alpha)\times\rho'[\alpha:=\beta]}(\mu\alpha\rho)'((\mu\alpha\rho)' \times \sigma')(\lambda x^{(\mu\alpha\rho)'}.\langle x, (\lambda x^{(\mu\alpha\rho)'}.x\mathrm{E}_\mu^+\sigma'\hat{s})x\rangle)\hat{t}\Big) \to_{\beta_\forall}^2\to_{\beta_\to}^2$
(setting $\hat{r} := (\lambda x^{(\mu\alpha\rho)'}.\langle x, (\lambda x^{(\mu\alpha\rho)'}.x\mathrm{E}_\mu^+\sigma'\hat{s})x\rangle)$)
$\hat{s}\Big(\hat{t}\mathrm{E}_\exists(\Lambda\beta\lambda y^{(\beta\to(\mu\alpha\rho)')\times\rho'[\alpha:=\beta]}.C_{\exists\beta.(\beta\to(\mu\alpha\rho)'\times\sigma')\times\rho'[\alpha:=\beta],\beta}\langle\lambda z^\beta.\hat{r}(y\mathsf{L}z), y\mathsf{R}\rangle)\Big) \to_{\beta_\exists}$
$\hat{s}\Big((\Lambda\beta\lambda y^{(\beta\to(\mu\alpha\rho)')\times\rho'[\alpha:=\beta]}.C_{\exists\beta.(\beta\to(\mu\alpha\rho)'\times\sigma')\times\rho'[\alpha:=\beta],\beta}\langle\lambda z^\beta.\hat{r}(y\mathsf{L}z), y\mathsf{R}\rangle)\tau'\langle j', t'\rangle\Big) \to_{\beta_\forall}\to_{\beta_\to}\to_{\beta_\times}^2$
$\hat{s}\Big(C_{\exists\beta.(\beta\to(\mu\alpha\rho)'\times\sigma')\times\rho'[\alpha:=\beta],\tau'}\langle\lambda z^{\tau'}.\hat{r}(j'z), t'\rangle\Big) = \hat{s}\Big(C_{\exists\alpha.(\alpha\to(\mu\alpha\rho)'\times\sigma')\times\rho',\tau'}\langle\lambda z^{\tau'}.\hat{r}(j'z), t'\rangle\Big) \to^2$
$\Big(\Lambda\alpha\lambda y^{(\alpha\to(\mu\alpha\rho)'\times\sigma')\times\rho'}.s'\alpha(\lambda u^\alpha.y\mathsf{L}u\mathsf{L})(\lambda u^\alpha.y\mathsf{L}u\mathsf{R})(y\mathsf{R})\Big)\tau'\langle\lambda z^{\tau'}.\hat{r}(j'z), t'\rangle \to_{\beta_\forall}\to_{\beta_\to}\to_{\beta_\times}^3$
$s'\tau'\Big(\lambda u^{\tau'}.(\lambda z^{\tau'}.\hat{r}(j'z))u\mathsf{L}\Big)\Big(\lambda u^{\tau'}.(\lambda z^{\tau'}.\hat{r}(j'z))u\mathsf{R}\Big)t' \to_{\beta_\to}^4\to_{\beta_\times}^2$
$s'\tau'\Big(\lambda u^{\tau'}.j'u\Big)\Big(\lambda u^{\tau'}.(\lambda x^{(\mu\alpha\rho)'}.x\mathrm{E}_\mu^+\sigma'\hat{s})(j'u)\Big)t' \to_{\beta_\to}$
$s'\tau'(\lambda x^{\tau'}j'x)(\lambda x^{\tau'}.(j'u)\mathrm{E}_\mu^+\sigma'\hat{s}))t' = \Big(s\tau(\lambda x^\tau.jx)(\lambda x^\tau.(jx)\mathrm{E}_\mu^+\sigma s)t\Big)'$

6.3 The dual **coMeIT** of **MeIT**

The system corresponding to $(\mathsf{lfp}^{\subseteq}\text{-I})$, $(\mathsf{lfp}^{\subseteq}\text{-E})$ and $(\mathsf{lfp}^{\subseteq}\text{-E}^+)$ is defined. The system varEMIT is embedded into it and it is embedded into its fragment without iteration and into coMePIT which motivated its definition, and its iterative fragment is embedded into eF.
The system coMeIT is dual to MeIT because it is motivated by the monotone operator $\Phi^{\subseteq}$ which is a construction dual to the definition of $\Phi^{\supseteq}$.

6.3.1 Definition

The (second) definition of typed terms and their free variables for system eF is extended by the following clauses:

(μ-I) If $t^{\forall\alpha.(\mu\alpha\rho\to\alpha)\to\rho} \in \Lambda$ then $(C_{\mu\alpha\rho}t)^{\mu\alpha\rho} \in \Lambda$ and $\mathsf{FV}(C_{\mu\alpha\rho}t) := \mathsf{FV}(t)$.

(μ-E) If $r^{\mu\alpha\rho} \in \Lambda$ and $s^{(\forall\alpha.(\sigma\to\alpha)\to\rho)\to\sigma} \in \Lambda$ (with $\alpha \notin \mathsf{FV}(\sigma)$), then $(rs)^\sigma \in \Lambda$ (also written as $r\mathrm{E}_\mu\sigma s$ or shorter $r\mathrm{E}_\mu s$) and $\mathsf{FV}(rs) := \mathsf{FV}(r) \cup \mathsf{FV}(s)$.

(μ-E$^+$) If $r^{\mu\alpha\rho} \in \Lambda$ and $s^{(\forall\alpha.(\mu\alpha\rho\times\sigma\to\alpha)\to\rho)\to\sigma} \in \Lambda$[5](with $\alpha \notin \mathsf{FV}(\sigma)$), then $(rs)^\sigma \in \Lambda$ (also written as $r\mathrm{E}_\mu^+\sigma s$ or shorter $r\mathrm{E}_\mu^+ s$) and $\mathsf{FV}(rs) := \mathsf{FV}(r) \cup \mathsf{FV}(s)$.

This definition is again compatible with the renaming convention for bound type variables.
Extend the recursive definition of $\mathsf{FTV}(r)$ for system eF by the following clauses:

(μ-I) $\mathsf{FTV}(C_{\mu\alpha\rho}t) := \mathsf{FV}(\mu\alpha\rho) \cup \mathsf{FTV}(t)$.

(μ-E) $\mathsf{FTV}(rs) := \mathsf{FTV}(r) \cup \mathsf{FTV}(s)$.

(μ-E$^+$) $\mathsf{FTV}(rs) := \mathsf{FTV}(r) \cup \mathsf{FTV}(s)$.

[5] It would be easy to avoid the use of $\times$ by stipulating $s^{(\forall\alpha.(\mu\alpha\rho\to\sigma\to\alpha)\to\rho)\to\sigma} \in \Lambda$ and changing (β_μ^+) accordingly. The embeddings into and from coMeIT would be less elegant, however.

The definition of $r[\vec{x}^{\vec{\rho}} := \vec{s}]$ for system eF together with the proof of $r^\rho[\vec{x}^{\vec{\rho}} := \vec{s}] : \rho$ and $\mathsf{FV}(r[\vec{x}^{\vec{\rho}} := \vec{s}]) = (\mathsf{FV}(r) \setminus \vec{x}^{\vec{\rho}}) \cup \bigcup\{\mathsf{FV}(s_i) | x_i^{\rho_i} \in \mathsf{FV}(r) \wedge x_i^{\rho_i} \neq x_j^{\rho_j} \text{ for } j < i\}$ (i. e., the statement of Lemma 2.10) is extended by the following clauses:

(μ-I) $(C_{\mu\alpha\rho}t)[\vec{x}^{\vec{\rho}} := \vec{s}] := C_{\mu\alpha\rho}t[\vec{x}^{\vec{\rho}} := \vec{s}]$. Proof obvious.

(μ-E) $(rs)[\vec{x}^{\vec{\rho}} := \vec{s}] := r[\vec{x}^{\vec{\rho}} := \vec{s}]\mathrm{E}_\mu s[\vec{x}^{\vec{\rho}} := \vec{s}]$. Proof obvious.

(μ-E$^+$) $(rs)[\vec{x}^{\vec{\rho}} := \vec{s}] := r[\vec{x}^{\vec{\rho}} := \vec{s}]\mathrm{E}_\mu^+ s[\vec{x}^{\vec{\rho}} := \vec{s}]$. Proof obvious.

The definition of $r[\vec{\alpha} := \vec{\sigma}]$ for system eF together with the proof of $r^\rho[\vec{\alpha} := \vec{\sigma}] : \rho[\vec{\alpha} := \vec{\sigma}]$ and $\mathsf{FV}(r[\vec{\alpha} := \vec{\sigma}]) = \{y^{\tau[\vec{\alpha}:=\vec{\sigma}]} | y^\tau \in \mathsf{FV}(r)\}$ is extended by the following clauses:

(μ-I) $(C_{\mu\alpha\rho}t)[\vec{\alpha} := \vec{\sigma}] := C_{(\mu\alpha\rho)[\vec{\alpha}:=\vec{\sigma}]}t[\vec{\alpha} := \vec{\sigma}]$. Proof obvious.

(μ-E) $(rs)[\vec{\alpha} := \vec{\sigma}] := r[\vec{\alpha} := \vec{\sigma}]\mathrm{E}_\mu s[\vec{\alpha} := \vec{\sigma}]$. Proof obvious.

(μ-E$^+$) $(rs)[\vec{\alpha} := \vec{\sigma}] := r[\vec{\alpha} := \vec{\sigma}]\mathrm{E}_\mu^+ s[\vec{\alpha} := \vec{\sigma}]$. Proof obvious.

The statements made in section 2.1.2 are valid in coMeIT, too.
Extend the inductive definition of $r\vec{s}$ for system eF as follows:

(μ-E) If $r\vec{s} : \mu\alpha\rho$ and $s : (\forall\alpha.(\sigma \to \alpha) \to \rho) \to \sigma$ (with $\alpha \notin \mathsf{FV}(\sigma)$), then $r(\vec{s}, s) := (r\vec{s})s$.

(μ-E$^+$) If $r\vec{s} : \mu\alpha\rho$ and $s : (\forall\alpha.(\mu\alpha\rho \times \sigma \to \alpha) \to \rho) \to \sigma$ (with $\alpha \notin \mathsf{FV}(\sigma)$), then $r(\vec{s}, s) := (r\vec{s})s$.

The statement of Lemma 2.23 holds also true for coMeIT.

Lemma 6.4 Every term has exactly one of the forms given in Lemma 2.44 or exactly one of the following forms:

(μ-I) $C_{\mu\alpha\rho}t$

(μ-R) $(C_{\mu\alpha\rho}t)s\vec{s}$, subsuming both eliminations for μ

Proof Induction on terms. □

Extend the inductive definition of the set NF for system eF by the following clause:

(μ-I) If $C_{\mu\alpha\rho}t$ is a term and $t \in \mathsf{NF}$, then $C_{\mu\alpha\rho}t \in \mathsf{NF}$.

The statements in section 2.1.3 hold again for coMeIT.
Define the relation $\mapsto$ as for system eF, but add the following clauses:

coMeIT

$$
\begin{array}{ll}
(\beta_\mu) & (C_{\mu\alpha\rho}t)\mathrm{E}_\mu\sigma s \mapsto s\Big(\Lambda\alpha\lambda z^{\sigma\to\alpha}.t\alpha(\lambda x^{\mu\alpha\rho}.z(x\mathrm{E}_\mu\sigma s))\Big) \\
(\beta_\mu^+) & (C_{\mu\alpha\rho}t)\mathrm{E}_\mu^+\sigma s \mapsto s\Big(\Lambda\alpha\lambda z^{\mu\alpha\rho\times\sigma\to\alpha}.t\alpha\Big(\lambda x^{\mu\alpha\rho}.z\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma s)x\rangle\Big)\Big)
\end{array}
$$

In both rules we require that $x^{\mu\alpha\rho} \notin \mathsf{FV}(s)$ and that z with the respective type is not in $\mathsf{FV}(s) \cup \mathsf{FV}(t)$ and we assume that $\alpha \notin \mathsf{FTV}(s) \cup \mathsf{FTV}(t)$.
As for system eF the objects on the right side are automatically terms if the respective object on the left is a term, and the types are equal.
The relation $\to_{whd}$ is defined as for system F and has the same properties as stated there.
Extend the inductive definition of the relation $\to$ for system eF by the congruence rules for μ and prove that if $r \to r'$, then r and r' have the same type and $\mathsf{FV}(r') \subseteq \mathsf{FV}(r)$.
Extend the inductive definition of WN for system eF by the following clauses:

(μ-I) If $C_{\mu\alpha\rho}t$ is a term and $t \in \mathsf{WN}$, then $C_{\mu\alpha\rho}t \in \mathsf{WN}$.

(β_μ) If $s\Big(\Lambda\alpha\lambda z^{\sigma\to\alpha}.t\alpha(\lambda x^{\mu\alpha\rho}.z(x\mathrm{E}_\mu\sigma s))\Big)\vec{s} \in \mathsf{WN}$, the side conditions of the (β_μ)-rule are met and $(C_{\mu\alpha\rho}t)\mathrm{E}_\mu\sigma s\vec{s}$ is a term, then $(C_{\mu\alpha\rho}t)\mathrm{E}_\mu\sigma s\vec{s} \in \mathsf{WN}$.

(β_μ^+) If $s\Big(\Lambda\alpha\lambda z^{\mu\alpha\rho\times\sigma\to\alpha}.t\alpha\Big(\lambda x^{\mu\alpha\rho}.z\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma s)x\rangle\Big)\Big)\vec{s} \in \mathsf{WN}$, the side conditions of the (β_μ^+)-rule are met and $(C_{\mu\alpha\rho}t)\mathrm{E}_\mu^+\sigma s\vec{s}$ is a term, then $(C_{\mu\alpha\rho}t)\mathrm{E}_\mu^+\sigma s\vec{s} \in \mathsf{WN}$.

Again definition by recursion on WN is admissible. Extend the recursive definition of the function Ω from WN to Λ and the simultaneous proof of $r \to^* \Omega(r) \in \mathsf{NF}$ for $r \in \mathsf{WN}$ by the following clauses:

(μ-I) $\Omega(C_{\mu\alpha\rho}t) := C_{\mu\alpha\rho}\Omega(t)$.

(β_μ) $\Omega((C_{\mu\alpha\rho}t)\mathrm{E}_\mu\sigma s\vec{s}) := \Omega\Big(s\Big(\Lambda\alpha\lambda z^{\sigma\to\alpha}.t\alpha(\lambda x^{\mu\alpha\rho}.z(x\mathrm{E}_\mu\sigma s))\Big)\vec{s}\Big)$.

(β_μ^+) $\Omega((C_{\mu\alpha\rho}m)\mathrm{E}_\mu^+\sigma s\vec{s}) := \Omega\Big(s\Big(\Lambda\alpha\lambda z^{\mu\alpha\rho\times\sigma\to\alpha}.t\alpha\Big(\lambda x^{\mu\alpha\rho}.z\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma s)x\rangle\Big)\Big)\vec{s}\Big)$.

The proofs are always obvious.
Extend the definition of SN for system eF by the following clauses:

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

(β_μ) If $s\Big(\Lambda\alpha\lambda z^{\sigma\to\alpha}.t\alpha(\lambda x^{\mu\alpha\rho}.z(x\mathrm{E}_\mu\sigma s))\Big)\vec{s} \in \mathsf{SN}$, the side conditions of the (β_μ)-rule are met and $(C_{\mu\alpha\rho}t)\mathrm{E}_\mu\sigma s\vec{s}$ is a term, then $(C_{\mu\alpha\rho}t)\mathrm{E}_\mu\sigma s\vec{s} \in \mathsf{SN}$.

(β_μ^+) If $s\Big(\Lambda\alpha\lambda z^{\mu\alpha\rho\times\sigma\to\alpha}.t\alpha\Big(\lambda x^{\mu\alpha\rho}.z\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma s)x\rangle\Big)\Big)\vec{s} \in \mathsf{SN}$, the side conditions of the (β_μ^+)-rule are met and $(C_{\mu\alpha\rho}t)\mathrm{E}_\mu^+\sigma s\vec{s}$ is a term, then $(C_{\mu\alpha\rho}t)\mathrm{E}_\mu^+\sigma s\vec{s} \in \mathsf{SN}$.

The only difference between the μ-clauses for SN and WN is the name of the defined set.
Every statement made in section 2.1.4 for system F (on nf, NF, wn, WN, Ω, SN and sn) is also true for coMeIT.

6.3.2 Embedding varEMIT in coMeIT

The motivation for this section is similar to the motivation for the embedding of $\mathsf{varIMIT}$ into MeIT. In the introduction to this chapter it was shown that ($\mathsf{lfp}^{\subseteq}$-I), ($\mathsf{lfp}^{\subseteq}$-E) and ($\mathsf{lfp}^{\subseteq}$-E$^+$) are equivalent to (lfp-I), (lfp-E) and (lfp-E$^+$) after replacing Φ by $\Phi^{\subseteq}$ in the latter three rules. If Φ is monotone, then $\Phi^{\subseteq} = \Phi$. Hence, by using monotonicity of Φ we may derive (lfp-I), (lfp-E) and (lfp-E$^+$) from ($\mathsf{lfp}^{\subseteq}$-I), ($\mathsf{lfp}^{\subseteq}$-E) and ($\mathsf{lfp}^{\subseteq}$-E$^+$), respectively, where $\mathsf{lfp}^{\subseteq}$ has been replaced by lfp in the latter three rules. Because monotonicity does not enter the proofs for the elimination rules and it is only used in a restricted way for the introduction rule, it is possible to produce an embedding of $\mathsf{varEMIT}$ into coMeIT out of the proofs. Of course, one has to check that reduction is also preserved.
Set $\rho' := \rho$ for every type ρ of $\mathsf{varEMIT}$. Define for every term r of system $\mathsf{varEMIT}$ the term r' of coMeIT by recursion on r and simultaneously prove that for $r : \rho$ we have $r' : \rho$ and $\mathsf{FTV}(r') = \mathsf{FTV}(r)$ and $\mathsf{FV}(r') = \mathsf{FV}(r)$ as follows:

(eF) The homomorphic term rules for eF.

(μ-I) $(C_{\mu\alpha\rho}m^{\forall\alpha.(\mu\alpha\rho\to\alpha)\to\rho[\alpha:=\mu\alpha\rho]\to\rho}t^{\rho[\alpha:=\mu\alpha\rho]})' := C_{\mu\alpha\rho}(\Lambda\alpha\lambda k^{\mu\alpha\rho\to\alpha}.m'\alpha k t')$

(μ-E) $(r^{\mu\alpha\rho}\mathrm{E}_\mu\sigma s^{\rho[\alpha:=\sigma]\to\sigma})' := r'\mathrm{E}_\mu\sigma(\lambda f^{\forall\alpha.(\sigma\to\alpha)\to\rho}.s'(f\sigma(\lambda y^\sigma y)))$

(μ-E$^+$) $(r^{\mu\alpha\rho}\mathrm{E}^+_\mu\sigma s^{\rho[\alpha:=\mu\alpha\rho\times\sigma]\to\sigma})' := r'\mathrm{E}^+_\mu\sigma(\lambda f^{\forall\alpha.(\mu\alpha\rho\times\sigma\to\alpha)\to\rho}.s'(f(\mu\alpha\rho\times\sigma)(\lambda y^{\mu\alpha\rho\times\sigma}y)))$

(The proofs are trivial.) Note that in both elimination cases the variable f is simply applied to the identity (after instantiation to a type).
The statement of Lemma 4.16 is true for this definition and is proved by induction on r.
That $-'$ is an embedding is proved by induction on $\to$. Only (β_μ) and (β^+_μ) deserve attention.
Check e. g. (β^+_μ):

$((C_{\mu\alpha\rho}mt)\mathrm{E}^+_\mu\sigma s)' =$
$C_{\mu\alpha\rho}(\Lambda\alpha\lambda k^{\mu\alpha\rho\to\alpha}.m'\alpha k t')\mathrm{E}^+_\mu\sigma(\lambda f^{\forall\alpha.(\mu\alpha\rho\times\sigma\to\alpha)\to\rho}.s'(f(\mu\alpha\rho\times\sigma)(\lambda y^{\mu\alpha\rho\times\sigma}y))) \mapsto_{\beta^+_\mu}$
$\Big(\text{setting } \hat{s} := (\lambda f^{\forall\alpha.(\mu\alpha\rho\times\sigma\to\alpha)\to\rho}.s'(f(\mu\alpha\rho\times\sigma)(\lambda y^{\mu\alpha\rho\times\sigma}y)))\Big)$
$\hat{s}\Big(\Lambda\alpha\lambda z^{\mu\alpha\rho\times\sigma\to\alpha}.(\Lambda\alpha\lambda k^{\mu\alpha\rho\to\alpha}.m'\alpha k t')\alpha(\lambda x^{\mu\alpha\rho}.z\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma\hat{s})x\rangle)) \to_{\beta_\forall}\to_{\beta_\to}$
$\hat{s}\Big(\Lambda\alpha\lambda z^{\mu\alpha\rho\times\sigma\to\alpha}.m'\alpha(\lambda x^{\mu\alpha\rho}.z\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma\hat{s})x\rangle)t'\Big) \to_{\beta_\to}\to_{\beta_\forall}\to_{\beta_\to}$
$s'\Big(m'(\mu\alpha\rho\times\sigma)(\lambda x^{\mu\alpha\rho}.(\lambda y^{\mu\alpha\rho\times\sigma}y)\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma\hat{s})x\rangle)t'\Big) \to_{\beta_\to}$
$s'\Big(m'(\mu\alpha\rho\times\sigma)(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma\hat{s})x\rangle)t'\Big) =$
$\Big(s\Big(m(\mu\alpha\rho\times\sigma)(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma s)x\rangle)t\Big)\Big)'$

6.3.3 Embedding coMeIT in coMePIT

This section is nothing but the syntactic analogue of the considerations which motivated the definition of coMeIT including the verification that reduction is preserved.
For every type ρ of system coMeIT define the type ρ' of system coMePIT by recursion on ρ and simultaneously prove that $\mathsf{FV}(\rho') = \mathsf{FV}(\rho)$ as follows:

(eF) The homomorphic type rules for eF.

(μ) $(\mu\alpha\rho)' := \mathsf{pos}(\mu\alpha\rho') = \mu\alpha\forall\beta.(\alpha\to\beta)\to\rho'[\alpha:=\beta]$ for $\beta\notin\{\alpha\}\cup\mathsf{FV}(\rho)$ which is a type of coMePIT because $\alpha\notin\{\beta\}\cup\mathsf{FV}(\rho'[\alpha:=\beta])$.

(The proofs were obvious.) This definition is compatible with the variable conventions for the systems coMeIT and coMePIT.

Lemma 6.5 $(\rho[\vec{\alpha}:=\vec{\sigma}])' = \rho'[\vec{\alpha}:=\vec{\sigma}']$.

Proof Induction on ρ. □

Define for every term r of system coMeIT the term r' of coMePIT by recursion on r and simultaneously prove that $r' : \rho'$ and $\mathsf{FTV}(r') = \mathsf{FTV}(r)$ and $\mathsf{FV}(r') = \{x^{\sigma'} | x^\sigma \in \mathsf{FV}(r)\}$ as follows:

(eF) The homomorphic term rules for eF.

(μ-I) $(C_{\mu\alpha\rho}t^{\forall\alpha.(\mu\alpha\rho\to\alpha)\to\rho})' := C_{(\mu\alpha\rho)'}t'$.

(μ-E) $(r^{\mu\alpha\rho}\mathrm{E}_\mu\sigma s^{(\forall\alpha.(\sigma\to\alpha)\to\rho)\to\sigma})' := r'\mathrm{E}_\mu\sigma s'$.

(μ-E$^+$) $(r^{\mu\alpha\rho}\mathrm{E}^+_\mu\sigma s^{(\forall\alpha.(\mu\alpha\rho\times\sigma\to\alpha)\to\rho)\to\sigma})' := r'\mathrm{E}^+_\mu\sigma s'$.

(The proofs are trivial except for $r' : \rho'$.) Obviously, $-'$ is the identity on the underlying untyped terms. (coMeIT has been designed such this trivial embedding works.)
Substitutivity (the statement of Lemma 4.11) is again true for this definition.
By induction on $\to$ show that $-'$ indeed is an embedding. The only cases of interest are (β_μ) and (β^+_μ). We only check (β^+_μ):

$((C_{\mu\alpha\rho}t)\mathrm{E}^+_\mu\sigma s)' = (C_{(\mu\alpha\rho)'}t')\mathrm{E}^+_\mu\sigma' s' \mapsto_{\beta^+_\mu}$
$s'\Big(\mathsf{map}_{\lambda\alpha\forall\beta.(\alpha\to\beta)\to\rho'}(\mu\alpha\rho)'((\mu\alpha\rho)'\times\sigma')(\lambda x^{(\mu\alpha\rho)'}.\langle x, (\lambda x^{(\mu\alpha\rho)'}.x\mathrm{E}^+_\mu\sigma' s')x\rangle)t'\Big) \to^2_{\beta_\forall}\to^3_{\beta_\to}$
$s'\Big(\Lambda\beta\lambda y^{(\mu\alpha\rho)'\times\sigma'\to\beta}.t'\beta\Big(\lambda x^{(\mu\alpha\rho)'}.y\langle x, (\lambda x^{(\mu\alpha\rho)'}.x\mathrm{E}^+_\mu\sigma' s')x\rangle\Big)\Big) =$
$\Big(s\Big(\Lambda\alpha\lambda z^{\mu\alpha\rho\times\sigma\to\alpha}.t\alpha\Big(\lambda x^{\mu\alpha\rho}.z\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma s)x\rangle\Big)\Big)\Big)'$

We see that the verification that $-'$ gives an embedding is completely straightforward because also the reduction rules for coMeIT were designed for that purpose. But note that due to the fact that β-reduction for μ in coMePIT is defined by help of map (and not only by $\underline{\mathsf{map}}$ and a substitution) we get some extra reductions which also show that an embedding of coMePIT in coMeIT cannot be given in an easy way.

6.3.4 Embedding coMeIT in coMeIT–it

Let coMeIT–it be the system which is derived from coMeIT by leaving out the term rule (μ-E) and the reduction rule (β_μ).
Set $\rho' := \rho$ and for every term r^ρ of system coMeIT define the term r' of system coMeIT–it by recursion on r and simultaneously prove that $r' : \rho$ and $\mathsf{FTV}(r') = \mathsf{FTV}(r)$ and $\mathsf{FV}(r') = \mathsf{FV}(r)$ as follows:

(eF) The homomorphic term rules for eF.

(μ-I) The homomorphic μ-introduction rule.

(μ-E) $(r^{\mu\alpha\rho}\mathrm{E}_\mu\sigma s)' := r'\mathrm{E}^+_\mu\sigma\hat{s}$ with

$$\hat{s} := \lambda z^{\forall\alpha.(\mu\alpha\rho\times\sigma\to\alpha)\to\rho}.s'(\Lambda\alpha\lambda y^{\sigma\to\alpha}.z\alpha(\lambda k^{\mu\alpha\rho\times\sigma}.y(k\mathsf{R}))),$$

where we assume that $\alpha\notin\mathsf{FV}(\sigma)$ and require $z\notin\mathsf{FV}(s)$.

(μ-E$^+$) The homomorphic μ^+-elimination rule.

(The proofs are obvious.) Substitutivity (the statement of Lemma 4.16) is true for this definition and is proved by induction on r. That $-'$ is an embedding is proved by induction on $\to$. Only the rule (β_μ) deserves attention.

$$\begin{array}{lcl}
(\beta_\mu)\!:\quad ((C_{\mu\alpha\rho}t)\mathrm{E}_\mu\sigma s)' & = & (C_{\mu\alpha\rho}t')\mathrm{E}^+_\mu\sigma\hat{s} \quad \text{with } \hat{s} \text{ as defined above} \\
& \mapsto & \hat{s}\Big(\Lambda\alpha\lambda z^{\mu\alpha\rho\times\sigma\to\alpha}.t'\alpha\Big(\lambda x^{\mu\alpha\rho}.z\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma\hat{s})x\rangle\Big)\Big) \\
& \to^3 & s'\Big(\Lambda\alpha\lambda y^{\sigma\to\alpha}.t'\alpha\Big(\lambda x^{\mu\alpha\rho}.(\lambda k^{\mu\alpha\rho\times\sigma}.y(k\mathsf{R}))\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma\hat{s})x\rangle\Big)\Big) \\
& \to^3 & s'\Big(\Lambda\alpha\lambda y^{\sigma\to\alpha}.t'\alpha\Big(\lambda x^{\mu\alpha\rho}.y(x\mathrm{E}^+_\mu\sigma\hat{s})\Big)\Big) \\
& = & \Big(s\Big(\Lambda\alpha\lambda z^{\sigma\to\alpha}.t\alpha(\lambda x^{\mu\alpha\rho}.z(x\mathrm{E}_\mu\sigma s))\Big)\Big)'
\end{array}$$

6.3.5 Embedding coMeIT–rec in eF

Let coMeIT–rec be the system which is derived from coMeIT by leaving out the term generation rule (μ-E$^+$) and the reduction rule (β^+_μ).
For every type ρ of system coMeIT–rec define the type ρ' of system eF by recursion on ρ and simultaneously prove that $\mathsf{FV}(\rho') = \mathsf{FV}(\rho)$ as follows:

(eF) The homomorphic type rules for eF.

(μ) $(\mu\alpha\rho)' := \forall\beta.((\forall\alpha.(\beta\to\alpha)\to\rho')\to\beta)\to\beta$ for $\beta\notin\{\alpha\}\cup\mathsf{FV}(\rho)$.

(The proofs were obvious.) This definition is compatible with the variable conventions for the systems coMeIT−rec and eF.

Lemma 6.6 $(\rho[\vec\alpha:=\vec\sigma])' = \rho'[\vec\alpha:=\vec\sigma']$.

Proof Induction on ρ. □

For every term r^ρ of system coMeIT−rec define the term r' of system eF by recursion on r and simultaneously prove that $r':\rho'$ and $\mathsf{FTV}(r') = \mathsf{FTV}(r)$ and $\mathsf{FV}(r') = \{x^{\sigma'} | x^\sigma \in \mathsf{FV}(r)\}$ as follows:

(eF) The homomorphic term rules for eF.

(μ-I) $(C_{\mu\alpha\rho}t)' := \Lambda\beta\lambda z^{(\forall\alpha.(\beta\to\alpha)\to\rho')\to\beta}.z\Big(\Lambda\alpha\lambda y^{\beta\to\alpha}.t'\alpha(\lambda x^{(\mu\alpha\rho)'}.y(x\beta z))\Big)$ for $\beta\notin\{\alpha\}\cup\mathsf{FV}(\rho)\cup\mathsf{FTV}(t)$ and $y,z\notin\mathsf{FV}(t)$.

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

(The proofs were always obvious.) This definition is again compatible with the variable conventions for coMeIT−rec and eF.
The statement of Lemma 4.11 is true for $-'$ and proved by induction on the terms. That $-'$ is an embedding is proved by induction on $\to$. Only (β_μ) deserves our attention:

$$\begin{aligned}((C_{\mu\alpha\rho}t)\mathrm{E}_\mu\sigma s)' &= \Big(\Lambda\beta\lambda z^{(\forall\alpha.(\beta\to\alpha)\to\rho')\to\beta}.z\Big(\Lambda\alpha\lambda y^{\beta\to\alpha}.t'\alpha(\lambda x^{(\mu\alpha\rho)'}.y(x\beta z))\Big)\Big)\sigma's'\\ &\to^2 s'\Big(\Lambda\alpha\lambda y^{\sigma'\to\alpha}.t'\alpha(\lambda x^{(\mu\alpha\rho)'}.y(x\sigma's'))\Big)\\ &= \Big(s\Big(\Lambda\alpha\lambda z^{\sigma\to\alpha}.t\alpha(\lambda x^{\mu\alpha\rho}.z(x\mathrm{E}_\mu\sigma s))\Big)\Big)'.\end{aligned}$$

There is another way to get an embedding of coMeIT−rec in eF which goes as follows: coMeIT embeds into varEMIT by composing the embeddings via coMePIT, ESMIT and EMIT. Obviously, one may leave the rule (μ-E$^+$) out of all these embeddings in order to have an embedding of coMeIT−rec in varEMIT−rec. Because there are also embeddings of varEMIT−rec in IT and of IT in eF we are done.

Chapter 7

Composing the embeddings

Let us first list all of the embeddings proved in the preceding chapters. In addition it is listed if a system is an instances of a class of systems or if it is a subsystem of a system (giving rise to a trivial embedding). We write $\preccurlyeq$ for "embeds into", $\in$ for "is an instance of" and $\leq$ for "is a subsystem of". We also write $\preccurlyeq$ and $\leq$ between a system and a class of systems if the relation holds for each of those systems. As expected we use $\subseteq$ and $\cap$ set-theoretically for classes of systems. (An asterisk marks the relation as interesting although it may be easy to establish. Two asterisks stress the importance.)

- F $\leq$ eF $\preccurlyeq^*$ F.
- eF $\leq$ IT $\preccurlyeq^*$ eF.
- eF $\leq$ eF+ex $\preccurlyeq^*$ eF.
- eF $\leq$ EMIT.
- eF $\leq$ ESMIT $\preccurlyeq^{**}$ EMIT.
- ESMITit $\subseteq$ ESMIT and ESMITrec $\subseteq$ ESMIT.
- eF+ex $\leq$ ESMIT+ex.
- eF $\leq$ varEMIT, EMIT $\preccurlyeq$ varEMIT, eF $\leq$ varEMIT–rec $\leq$ varEMIT and varEMIT–rec $\preccurlyeq^*$ IT.
- eF $\leq$ IMIT, eF $\leq$ IMIT–rec $\leq$ IMIT, eF $\leq$ IMIT–it $\leq$ IMIT.
- eF $\leq$ EMIT–it $\leq$ EMIT, eF+ex $\leq$ ISMIT+ex and IMIT $\leq$ IMIT+ex.
- The following embeddings were only sketched (in section 4.5.2): IMIT $\preccurlyeq^*$ IMIT–it and EMIT $\preccurlyeq^*$ EMIT–it. Auxiliary were: NISPIT+ex $\in$ ISMIT+ex, ISMIT+ex $\preccurlyeq$ IMIT+ex, IMIT+ex $\preccurlyeq$ IMIT, MeIT–it $\preccurlyeq$ IMIT–it and coMeIT–it $\preccurlyeq$ EMIT–it.
- eF $\leq$ ISMIT $\preccurlyeq^*$ IMIT.
- eF $\leq$ varIMIT and IMIT $\preccurlyeq$ varIMIT.
- PIT = PITit $\in^{**}$ ESMITit $\cap$ ISMIT and PITrec $\in$ ESMITrec.
- SPIT $\leq$ PIT and SPIT $\in$ ESMITit.
- PIT+ex $\in$ ESMIT+ex and SPIT+ex $\in$ ESMIT+ex.

- NIPIT $\leq$ PIT, NIPIT $\in^*$ ESMITit $\cap$ ESMITrec.
- NISPIT $\leq$ SPIT, NISPIT $\in$ ESMITit and NISPIT+ex $\leq$ SPIT+ex.
- MePIT $\in$ ESMIT+ex and MePIT $\leq$ NISPIT+ex.
- coMePIT $\in$ ESMIT and coMePIT $\leq$ NIPIT.
- eF $\leq$ MeIT and varIMIT $\preccurlyeq^{**}$ MeIT.
- eF $\leq$ MeIT–it $\leq$ MeIT and MeIT $\preccurlyeq^*$ MeIT–it.
- eF $\leq$ MeIT–rec $\leq$ MeIT and MeIT–rec $\preccurlyeq$ eF.
- eF $\leq$ UVIT and MeIT $\preccurlyeq$ UVIT $\preccurlyeq^*$ MePIT.
- eF $\leq$ coMeIT and varEMIT $\preccurlyeq^{**}$ coMeIT $\preccurlyeq^*$ coMePIT.
- eF $\leq$ coMeIT–it $\leq$ coMeIT and coMeIT $\preccurlyeq^*$ coMeIT–it.
- eF $\leq$ coMeIT–rec $\leq$ coMeIT and coMeIT–rec $\preccurlyeq$ eF.

We may now prove the part of the theorem in chapter 1 pertaining to embeddings:
The systems of class I embed into each other: F $\leq$ eF $\preccurlyeq$ F. The other systems are defined as extensions of eF. Hence, they only have to be embedded into eF: This has been shown directly for IT, eF+ex, varEMIT–rec, MeIT–rec and coMeIT–rec. For IMIT–rec we argue as follows: In the proof of IMIT $\preccurlyeq$ varIMIT $\preccurlyeq$ MeIT one may leave out (μ-E$^+$) everwhere and thus gets IMIT–rec $\preccurlyeq$ MeIT–rec. Because MeIT–rec $\preccurlyeq$ eF, we are done.
The systems of class IIa embed into one of class III: NISPIT and SPIT are subsystems of PIT and hence embed into it. Moreover, they are extensions of eF. Hence, every system of class I embeds into them.
The systems of class IIb are all extensions of eF. They embed into one of the systems of class III: ESMIT $\preccurlyeq$ EMIT, hence trivially also ESMITit $\preccurlyeq$ EMIT and ESMITrec $\preccurlyeq$ EMIT. ISMIT $\preccurlyeq$ IMIT. Let us define the system EMIT+ex as the obvious extension of EMIT by $\exists$. Clearly, the proof that ESMIT $\preccurlyeq$ EMIT extends to ESMIT+ex $\preccurlyeq$ EMIT+ex. Also, the proof that eF+ex $\preccurlyeq$ eF extends to EMIT+ex $\preccurlyeq$ EMIT. Therefore, ESMIT+ex $\preccurlyeq$ EMIT. ISMIT+ex $\preccurlyeq$ IMIT+ex has already been listed.
The systems of class III embed into each other: NIPIT $\leq$ PIT $\in$ ISMIT. Also NIPIT $\leq$ PITrec $\in$ ISMIT because NIPIT does neither use iteration nor recursion in the map terms and because the height measure h does not only establish that PITrec $\in$ ESMIT but also PITrec $\in$ ISMIT.
ISMIT $\preccurlyeq$ IMIT $\preccurlyeq$ varIMIT $\preccurlyeq$ MeIT $\preccurlyeq$ UVIT $\preccurlyeq$ MePIT $\leq$ NISPIT+ex $\leq$ SPIT+ex $\leq$ PIT+ex $\in$ ESMIT+ex $\preccurlyeq$ EMIT+ex $\preccurlyeq$ EMIT $\preccurlyeq$ varEMIT $\preccurlyeq$ coMeIT $\preccurlyeq$ coMePIT $\leq$ NIPIT.
Hence, the concrete members of this chain (not each of ISMIT and ESMIT+ex) embed into each other and we are done with NIPIT, PIT=PITit, PITrec, IMIT, varIMIT, MeIT, UVIT, MePIT, NISPIT+ex, SPIT+ex, PIT+ex, EMIT, varEMIT, coMeIT and coMePIT.
Therefore only IMIT+ex, MeIT–it, coMeIT–it, EMIT–it and IMIT–it remain to be dealt with. But IMIT $\leq$ IMIT+ex $\preccurlyeq$ IMIT, MeIT–it $\leq$ MeIT $\preccurlyeq$ MeIT–it and coMeIT–it $\leq$ coMeIT $\preccurlyeq$ coMeIT–it. These embeddings were used to etablish the same for IMIT–it and EMIT–it in section 4.5.2.
Confluence will be addressed in the next chapter and strong normalization in chapter 9. □

As a main consequence of the theorem I consider the equivalence with respect to β-reduction of EMIT and IMIT. Hence, the two ways of fixing monotonicity witnesses to the term rules for $\mu\alpha\rho$ give essentially the same. An interesting technical detail is the embedding of varEMIT into EMIT and of $\mathsf{varIMIT}$ into IMIT, hence showing that the relaxation of the monotonicity assumption does not add to the capability of expressing algorithms[1].

As a consequence we also get e. g. $\mathsf{NISPIT{+}ex} \preccurlyeq \mathsf{NIPIT}$ and $\mathsf{NIPIT} \preccurlyeq \mathsf{NISPIT{+}ex}$ which is not at all obvious! One might think that the first embedding is furnished by simply coding the existential quantifier as in $\mathsf{eF{+}ex} \preccurlyeq \mathsf{eF}$. A coding is gained (note that it does not preserve strict positivity and hence has target NIPIT), but it does not preserve reduction because the map term is not transformed to the map term for the transformed type: Let $-'$ be the encoding. Let $\rho := \exists\gamma\tau$ for some γ and τ such that $\alpha \in \mathsf{FV}(\rho)$. We have to show e. g. that

$$(C_{\mu\alpha\rho} t \mathrm{E}_\mu \sigma s)' \to^+ \Big(s\Big(\mathsf{map}_{\lambda\alpha\rho}(\mu\alpha\rho)\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma s)t\Big)\Big)'.$$

We have that

$$(C_{\mu\alpha\rho} t \mathrm{E}_\mu \sigma s)' = C_{\mu\alpha\rho'} t' \mathrm{E}_\mu \sigma' s' \mapsto s'\Big(\mathsf{map}_{\lambda\alpha\rho'}(\mu\alpha\rho')\sigma'(\lambda x^{\mu\alpha\rho'}.x\mathrm{E}_\mu\sigma' s')t'\Big)$$

and

$$\Big(s\Big(\mathsf{map}_{\lambda\alpha\rho}(\mu\alpha\rho)\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma s)t\Big)\Big)' = s'\Big((\mathsf{map}_{\lambda\alpha\rho})'(\mu\alpha\rho')\sigma'(\lambda x^{\mu\alpha\rho'}.x\mathrm{E}_\mu\sigma' s')t'\Big).$$

By looking at the definitions we see that $\mathsf{map}_{\lambda\alpha\rho'} \neq (\mathsf{map}_{\lambda\alpha\rho})'$ (the first corresponding $\underline{\mathsf{map}}$ starts with Λ-abstraction, the $\underline{\mathsf{map}}$ corresponding to the second term is a term of the form $x\vec{r}$). Because $\mathsf{map}_{\lambda\alpha\rho'} \in \mathsf{NF}$, also $\mathsf{map}_{\lambda\alpha\rho'} \not\to^* (\mathsf{map}_{\lambda\alpha\rho})'$. A second glance at the definitions reveals that also except in trivial cases

$$s'\Big(\mathsf{map}_{\lambda\alpha\rho'}(\mu\alpha\rho')\sigma'(\lambda x^{\mu\alpha\rho'}.x\mathrm{E}_\mu\sigma' s')t'\Big) \not\to^* s'\Big((\mathsf{map}_{\lambda\alpha\rho})'(\mu\alpha\rho')\sigma'(\lambda x^{\mu\alpha\rho'}.x\mathrm{E}_\mu\sigma' s')t'\Big).$$

We conclude that type-changing coding in general does not give embeddings for systems of selected monotone inductive types because the selected monotonicity witnesses for the encoded types are not the encoded witnesses.

Another interesting consequence of the theorem is $\mathsf{PIT} \preccurlyeq \mathsf{NIPIT}$. Hence, even non-strict positivity does not allow to make essential (in the sense of getting algorithms which cannot be simulated) use of interleaving.

Open questions remain, most importantly whether $\mathsf{SPIT} \preccurlyeq \mathsf{NISPIT}$ or $\mathsf{NIPIT} \preccurlyeq \mathsf{SPIT}$. If both were true, then class IIa could be merged in class III. In order to show both embeddings it is obviously enough to show that NIPIT embeds into NISPIT. In some sense it is the question whether the intuitionistic theory of positive inductive definitions may be reduced to the intuitionistic theory of strictly positive inductive definitions which has been answered positively by Buchholz in [BFPS81], chapter IV §5. But the analogy is not too close: Buchholz' systems do not have the second-order universal quantification because the aim is a proof-theoretic reduction. On the other side, there is no account of primitive recursion and no notion of β-reduction. Nevertheless it might be possible to get $\mathsf{NIPIT} \preccurlyeq \mathsf{NISPIT}$ from an analysis of the proof in [BFPS81]. If $\mathsf{NIPIT} \not\preccurlyeq \mathsf{NISPIT}$ then a proof of the non-embeddability would probably be even harder.

[1]Contrast this with the non-normalizing variant of IMIT considered in section 4.7.1.

Chapter 8

Confluence

A binary relation $\to$ (with transitive and reflexive closure $\to^*$) on a set M is locally confluent iff for every r, r' and r'' in M with $r \to r'$ and $r \to r''$ there is an $r''' \in M$ such that $r' \to^* r'''$ and $r'' \to^* r'''$. $\to$ is confluent iff $\to^*$ is locally confluent. (Note that $\to^{**}=\to^*$.) The major consequence of confluence is that there is at most one normal form of any term. If the system is also normalizing this gives decidability of the equality induced by $\to$. Our goal is to establish confluence for all the systems which have been introduced so far. The proof will use a variant of Masako Takahashi's method [Tak95] which in case of higher-order rewrite systems has been worked out in [vR96] and for Orthogonal Pattern Rewrite Systems in [MN98]. It has to be admitted that the variation on the method is not essential for getting the result. Also the restriction to typed terms is not necessary. However, only typed terms are considered in this thesis.
Note that there are also proofs of confluence via the computability predicate method presented in the next chapter for proving strong normalization (see e. g. [Kol85] where predicates of monovaluedness are studied). For the systems of this thesis the method has the drawback of reflecting the expressive power of the systems (which therefore includes intuitionistic second-order arithmetic) although the proof by complete superdevelopments shown here is an easy syntactic analysis.

8.1 Confluence of system F

Define the relation $\twoheadrightarrow$ of parallel reduction between terms[1] by induction and simultaneously prove that $r \twoheadrightarrow r' \Rightarrow r \to^* r'$ as follows:

(V) $x^\rho \twoheadrightarrow x^\rho$.

($\to$-I) If $r \twoheadrightarrow r'$, then $\lambda x^\rho r \twoheadrightarrow \lambda x^\rho r'$.

($\to$-E) If $r \twoheadrightarrow r'$ and $s \twoheadrightarrow s'$, then $r\mathrm{E}_\to s \twoheadrightarrow r'\mathrm{E}_\to s'$. (We use the notation of section 2.2.2.)

($\beta_\to$) If $r \twoheadrightarrow \lambda x^\rho r'$ and $s^\rho \twoheadrightarrow s'$, then $r\mathrm{E}_\to s \twoheadrightarrow r'[x^\rho := s']$.

($\forall$-I) If $r \twoheadrightarrow r'$, then $\Lambda\alpha r \twoheadrightarrow \Lambda\alpha r'$.

($\forall$-E) If $r \twoheadrightarrow r'$, then $r\mathrm{E}_\forall\sigma \twoheadrightarrow r'\mathrm{E}_\forall\sigma$.

[1]We adopt the convention of footnote 4 on p. 19. As for $\to$ a more careful formulation would clarify that the well-formedness of r' is always a consequence of the well-formedness of r in every clause which again gives an operational reading to $\twoheadrightarrow$. See also the discussion in section 2.1.4.

($\beta_\forall$) If $r \twoheadrightarrow \Lambda\alpha r'$, then $r\mathrm{E}_\forall\sigma \twoheadrightarrow r'[\alpha := \sigma]$.

(The proofs are omitted.)
Define for every term r the complete superdevelopment r^* of r as a term of the same type as r by recursion on terms and simultaneously prove that $r \twoheadrightarrow r^*$ as follows:

(V) $(x^\rho)^* := x^\rho$.

($\to$-I) $(\lambda x^\rho r)^* := \lambda x^\rho r^*$.

($\to$-E) $(r\mathrm{E}_\to s)^* := r^*\hat{\mathrm{E}}_\to s^*$. (See section 2.1.4 for the notation $\hat{\mathrm{E}}_\to$.)

($\forall$-I) $(\Lambda\alpha r)^* := \Lambda\alpha r^*$. (Note that $\Lambda\alpha r^*$ is well-formed due to $\mathsf{FV}(r^*) \subseteq \mathsf{FV}(r)$ which follows from the additional claim.)

($\forall$-E) $r\mathrm{E}_\forall\sigma := r^*\hat{\mathrm{E}}_\forall\sigma$. (See section 2.1.4 for the notation $\hat{\mathrm{E}}_\forall$.)

(The proofs are again omitted.)
Let us consider two essential examples of complete superdevelopments (omitting types everywhere):

$((\lambda xx)(\lambda yr)s)^* = r^*[y := s^*]$

$((\lambda x\lambda yr)st)^* = r^*[x, y := s^*, t^*]$, if $x \neq y$

We see that even created redices are contracted during the formation of the complete superdevelopment. Of course, we do not contract every created redex, e. g.

$$((\lambda x.xy)(\lambda zz))^* = (xy)[x := \lambda zz] = (\lambda zz)y.$$

The name superdevelopment was coined by van Raamsdonk [vR93]. I just mention that r^* is a maximal superdevelopment of r, if one extends van Raamsdonk's notion to system F properly. Below we will see that r^* is a maximal reduct with respect to $\twoheadrightarrow$. Nevertheless, $\twoheadrightarrow$ is not the superdevelopment relation ($\twoheadrightarrow$ is stronger) because in the definition of superdevelopments it is allowed that the term which is substituted via the ($\beta_\to$)-rule gets developped differently for the different free occurrences of x in r (which is obviously impossible in the definition of $\twoheadrightarrow$). One would get maximal developments if one replaced the elimination clauses of the definition of r^* in the following way:

($\to$-E) $((\lambda x^\rho t)s)^* := t^*[x^\rho := s^*]$, and $(rs)^* := r^*s^*$, if r is not a λ-abstraction.

($\forall$-E) $(\Lambda\alpha t)\sigma)^* := t^*[\alpha := \sigma]$, and $(r\sigma)^* := r^*\sigma$, if r is not a Λ-abstraction.

Deciding the cases according to the shape of r and not that of r^* makes the difference. This definition would fit to an alternate notion of parallel reduction by replacing the defining rules of $\twoheadrightarrow$ as follows:

($\beta_\to$) If $r \twoheadrightarrow r'$ and $s \twoheadrightarrow s'$, then $(\lambda x^\rho r)s \twoheadrightarrow r'[x^\rho := s']$.

($\beta_\forall$) If $r \twoheadrightarrow r'$, then $(\Lambda\alpha r)\sigma \twoheadrightarrow r'[\alpha := \sigma]$.

These definitions are the basis of the Takahashi method for proving confluence. My definitions give more flexibility to the method. This will not become clear from this thesis, however.

Lemma 8.1 (Reflexivity of $\twoheadrightarrow$) $r \twoheadrightarrow r$.

Proof Induction on r. (The rules $(\beta_\to)$ and $(\beta_\forall)$ are not needed here.) □

Corollary 8.2 $\to\subseteq\twoheadrightarrow$.

Proof Induction on $\to$ using reflexivity almost everywhere. □

Corollary 8.3 $\to^*=\twoheadrightarrow^*$.

Proof Use $\twoheadrightarrow\subseteq\to^*$ and $\to^{**}=\to^*$. □

Lemma 8.4 If $r \twoheadrightarrow r'$ and $\vec{s} \twoheadrightarrow \vec{s}'$ (i. e., $s_i \twoheadrightarrow s_i'$ for all i), then $r[\vec{x}^{\vec{\rho}} := \vec{s}] \twoheadrightarrow r'[\vec{x}^{\vec{\rho}} := \vec{s}']$.

Proof Induction on $r \twoheadrightarrow r'$. (V) Case $x^\rho \twoheadrightarrow x^\rho$. Obvious.
$(\to\text{-I})$ Case $\lambda x^\rho r \twoheadrightarrow \lambda x^\rho r'$. Easy by induction hypothesis.
$(\to\text{-E})$ Case $r\mathrm{E}_\to s \twoheadrightarrow r'\mathrm{E}_\to s'$. By induction hypothesis.
$(\beta_\to)$ Case $r\mathrm{E}_\to s \twoheadrightarrow r'[x^\rho := s']$. We may assume that $x^\rho \notin \vec{x}^{\vec{\rho}} \cup \mathsf{FV}(\vec{s})$. Therefore by induction hypothesis $r[\vec{x}^{\vec{\rho}} := \vec{s}] \twoheadrightarrow \lambda x^\rho.r'[\vec{x}^{\vec{\rho}} := \vec{s}']$ and $s[\vec{x}^{\vec{\rho}} := \vec{s}] \twoheadrightarrow s'[\vec{x}^{\vec{\rho}} := \vec{s}']$. Hence

$$(r\mathrm{E}_\to s)[\vec{x}^{\vec{\rho}} := \vec{s}] \twoheadrightarrow (r'[\vec{x}^{\vec{\rho}} := \vec{s}'])[x^\rho := s'[\vec{x}^{\vec{\rho}} := \vec{s}']] = (r'[x^\rho := s'])[\vec{x}^{\vec{\rho}} := \vec{s}']$$

by Lemma 2.13.
$(\forall\text{-I})$ Case $\Lambda\alpha r \twoheadrightarrow \Lambda\alpha r'$. We may assume that $\alpha \notin \mathsf{FTV}(\vec{s})$. By Lemma 2.30, $\alpha \notin \mathsf{FTV}(\vec{s}')$. The claim now follows from the induction hypothesis.
$(\forall\text{-E})$ Case $r\mathrm{E}_\forall\sigma \twoheadrightarrow r'\mathrm{E}_\forall\sigma$. By induction hypothesis.
$(\beta_\forall)$ Case $r\mathrm{E}_\forall\sigma \twoheadrightarrow r'[\alpha := \sigma]$. We may assume that $\alpha \notin \mathsf{FTV}(\vec{s}')$. Therefore by induction hypothesis $r[\vec{x}^{\vec{\rho}} := \vec{s}] \twoheadrightarrow \Lambda\alpha r'[\vec{x}^{\vec{\rho}} := \vec{s}']$. Hence, $(r\mathrm{E}_\forall\sigma)[\vec{x}^{\vec{\rho}} := \vec{s}] \twoheadrightarrow (r'[\vec{x}^{\vec{\rho}} := \vec{s}'])[\alpha := \sigma]$. Due to $\mathsf{FV}(\vec{\rho}) \subseteq \mathsf{FTV}(\vec{s}')$ (by Lemma 2.8), $\alpha \notin \mathsf{FV}(\vec{\rho})$. Because $\Lambda\alpha r'$ is well-formed, we have the implication $x^\rho \in \mathsf{FV}(r') \Rightarrow \alpha \notin \mathsf{FV}(\rho)$. Therefore Corollary 2.21 applies and gives that

$$(r'[\vec{x}^{\vec{\rho}} := \vec{s}'])[\alpha := \sigma] = r'[\alpha := \sigma][\vec{x}^{\vec{\rho}} := \vec{s}'[\alpha := \sigma]] = r'[\alpha := \sigma][\vec{x}^{\vec{\rho}} := \vec{s}']$$

(as $\alpha \notin \mathsf{FTV}(\vec{s}')$). □

Lemma 8.5 If $r \twoheadrightarrow r'$, then $r[\vec{\alpha} := \vec{\sigma}] \twoheadrightarrow r'[\vec{\alpha} := \vec{\sigma}]$.

Proof Induction on $r \twoheadrightarrow r'$. The cases (V), $(\to\text{-I})$, $(\to\text{-E})$, $(\forall\text{-I})$ and $(\forall\text{-E})$ are as easy as for the preceding lemma. The case $(\beta_\forall)$ uses Lemma 2.18.
Case $(\beta_\to)$: By induction hypothesis $r[\vec{\alpha} := \vec{\sigma}] \twoheadrightarrow (\lambda x^\rho r')[\vec{\alpha} := \vec{\sigma}]$. We may assume that $x^\tau \in \mathsf{FV}(r') \Rightarrow \tau = \rho$. Hence, $(\lambda x^\rho r')[\vec{\alpha} := \vec{\sigma}] = \lambda x^{\rho[\vec{\alpha}:=\vec{\sigma}]}.r'[\vec{\alpha} := \vec{\sigma}]$. By induction hypothesis $s[\vec{\alpha} := \vec{\sigma}] \twoheadrightarrow s'[\vec{\alpha} := \vec{\sigma}]$. Therefore

$$(rs)[\vec{\alpha} := \vec{\sigma}] \twoheadrightarrow (r'[\vec{\alpha} := \vec{\sigma}])[x^{\rho[\vec{\alpha}:=\vec{\sigma}]} := s'[\vec{\alpha} := \vec{\sigma}]].$$

This last term equals $r'[x^\rho := s'][\vec{\alpha} := \vec{\sigma}]$ by Corollary 2.22 (to the interchange law for substitution). □

Obviously, the parallel reduction relation $\twoheadrightarrow$ is not deterministic, i. e., given a term r there may be several terms r' such that $r \twoheadrightarrow r'$. If r is not an elimination there is only one clause in the definition which allows to derive $r \twoheadrightarrow r'$ for any r'. This immediately leads to the following

Lemma 8.6 (V) If $x \twoheadrightarrow t$, then $t = x$.

$(\to\text{-I})$ If $\lambda x^\rho r \twoheadrightarrow t$, then $t = \lambda x^\rho r'$ with $r \twoheadrightarrow r'$.

$(\forall\text{-I})$ If $\Lambda\alpha r \twoheadrightarrow t$, then $t = \Lambda\alpha r'$ with $r \twoheadrightarrow r'$.

Proof By induction on $\twoheadrightarrow$ without using the induction hypothesis. □

The following lemma is the key to proving confluence. It states that the complete superdevelopment r^* of a term r is not only a parallel reduct of r (as is shown simultaneously with the definition of r^*) but a parallel reduct of every parallel reduct r' of r. In this sense r^* is the maximal parallel reduct of r.

Lemma 8.7 (Maximality of r^*) If $r \twoheadrightarrow r'$, then $r' \twoheadrightarrow r^*$.

Proof Induction on $r \twoheadrightarrow r'$. The cases (V), $(\to\text{-I})$ and $(\forall\text{-I})$ are very easy.
$(\to\text{-E})$ Case $r\mathrm{E}_\to s \twoheadrightarrow r'\mathrm{E}_\to s'$. By induction hypothesis $r' \twoheadrightarrow r^*$ and $s' \twoheadrightarrow s^*$. If $r^* = \lambda x^\rho t$, then $r's' \twoheadrightarrow t[x^\rho := s^*] = (rs)^*$ by the rule $(\beta_\to)$. If r^* is not a λ-abstraction, then by rule $(\to\text{-E})$ $r's' \twoheadrightarrow r^*s^* = (rs)^*$.
$(\beta_\to)$ Case $r\mathrm{E}_\to s \twoheadrightarrow r'[x^\rho := s']$. By induction hypothesis $\lambda x^\rho r' \twoheadrightarrow r^*$ and $s' \twoheadrightarrow s^*$. By the preceding lemma $r^* = \lambda x^\rho r''$ with $r' \twoheadrightarrow r''$. Hence $(rs)^* = r''[x^\rho := s^*]$ and by Lemma 8.4 $r'[x^\rho := s'] \twoheadrightarrow r''[x^\rho := s^*]$.
The rules $(\forall\text{-E})$ and $(\beta_\forall)$ are treated similarly (where Lemma 8.5 is used instead of Lemma 8.4). □

Corollary 8.8 (“Diamond property” for $\twoheadrightarrow$) If $r \twoheadrightarrow r'$ and $r \twoheadrightarrow r''$, then there is a term r''' such that $r' \twoheadrightarrow r'''$ and $r'' \twoheadrightarrow r'''$.

Proof Set $r''' := r^*$. □

The main advantage of Takahashi's method over the traditional approach by parallel reduction seems to be the fact that r''' is identified as a term r^* which does not depend on r' and r''.
Define the binary relations $\twoheadrightarrow^n$, $n \in \mathbb{N}$, by recursion on n as follows:

$$r \twoheadrightarrow^0 s :\Leftrightarrow r = s \quad \text{and} \quad r \twoheadrightarrow^{n+1} s :\Leftrightarrow \exists t.r \twoheadrightarrow t \wedge t \twoheadrightarrow^n s.$$

Obviously, $r \twoheadrightarrow^* r' \Leftrightarrow \exists n.r \twoheadrightarrow^n r'$, and if $r \twoheadrightarrow^n r'$, then $r \twoheadrightarrow^m r'$ for all $m > n$.

Corollary 8.9 If $r \twoheadrightarrow^m r'$ and $r \twoheadrightarrow^n r''$, then there is a term r''' such that $r' \twoheadrightarrow^n r'''$ and $r'' \twoheadrightarrow^m r'''$.

Proof Induction on $m+n$. Trivial for $m = 0$ or $n = 0$. Otherwise there are terms s' and s'' such that $r \twoheadrightarrow s' \twoheadrightarrow^{m-1} r'$ and $r \twoheadrightarrow s'' \twoheadrightarrow^{n-1} r''$. By the preceding corollary, there is a term s (viz. r^*) such that $s' \twoheadrightarrow s$ and $s'' \twoheadrightarrow s$. $m-1+1 < m+n$, hence by induction hypothesis there is a term t such that $r' \twoheadrightarrow t$ and $s \twoheadrightarrow^{m-1} t$. $m-1+1+n-1 < m+n$, hence by induction hypothesis (applied to the term s'') there is a term r''' such that $t \twoheadrightarrow^{n-1} r'''$ and $r'' \twoheadrightarrow^{m-1+1} r'''$. Therefore also $r' \twoheadrightarrow^n r'''$. (Of course, it would be easier to draw a big rectangle built up from mn small rectangles derived by applying the “diamond property” that many times.) □

Corollary 8.10 (Confluence) $\to$ is confluent.

Proof We have to show that $\to^*=\twoheadrightarrow^*$ is locally confluent. This follows immediately from the preceding corollary. □

8.2 Confluence of the other systems

Unfortunately there is no result on preservation of confluence via embeddings. It is clear from the definition that even taking a subset of reduction rules may destroy confluence[2]. Of course, the confluence proof given in the previous section is also modular in the sense which allows us to appeal to the meta-convention on extensions introduced in section 2.2.3 (more precisely the extension thereof which will be introduced on p. 122).
Only the confluence proof for ESMIT will be given in the sequel. It will turn out that stratification is not needed, that the principles modelled by the system are not reflected and that compatibility with substitution of $\mathsf{map}_{\lambda\alpha\rho}$ is the only feature of ESMIT entering the arguments.
Extend the definition of the relation $\twoheadrightarrow$ of parallel reduction for system F to ESMIT and simultaneously prove that $r \twoheadrightarrow r' \Rightarrow r \to^* r'$ as follows:

(0-E) If $r \twoheadrightarrow r'$, then $r\mathrm{E}_0\rho \twoheadrightarrow r'\mathrm{E}_0\rho$.

(1-I) $\mathsf{IN1} \twoheadrightarrow \mathsf{IN1}$.

($\times$-I) If $r \twoheadrightarrow r'$ and $s \twoheadrightarrow s'$, then $\langle r, s\rangle \twoheadrightarrow \langle r', s'\rangle$.

($\times$-E) If $r \twoheadrightarrow r'$, then $r\mathsf{L} \twoheadrightarrow r'\mathsf{L}$ and $r\mathsf{R} \twoheadrightarrow r'\mathsf{R}$.

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

(+-I) If $r \twoheadrightarrow r'$, then $\mathsf{INL}_\sigma r \twoheadrightarrow \mathsf{INL}_\sigma r'$ and $\mathsf{INR}_\rho r \twoheadrightarrow \mathsf{INR}_\rho r'$.

(+-E) If $r \twoheadrightarrow r'$, $s \twoheadrightarrow s'$ and $t \twoheadrightarrow t'$, then $r\mathrm{E}_+ st \twoheadrightarrow r'\mathrm{E}_+ s't'$.

(β_+) If $r \twoheadrightarrow \mathsf{INL}_\sigma r'$ and $s \twoheadrightarrow s'$, then $r\mathrm{E}_+ st \twoheadrightarrow s'r'$. If $r \twoheadrightarrow \mathsf{INR}_\rho r'$ and $t \twoheadrightarrow t'$, then $r\mathrm{E}_+ st \twoheadrightarrow t'r'$.

(μ-I) If $t \twoheadrightarrow t'$, then $C_{\mu\alpha\rho}t \twoheadrightarrow C_{\mu\alpha\rho}t'$.

(μ-E) If $r \twoheadrightarrow r'$ and $s \twoheadrightarrow s'$, then $r\mathrm{E}_\mu s \twoheadrightarrow r'\mathrm{E}_\mu s'$.

(β_μ) If $r \twoheadrightarrow C_{\mu\alpha\rho}t$ and $s \twoheadrightarrow s'$, then $r\mathrm{E}_\mu \sigma s \twoheadrightarrow s'\Big(\mathsf{map}_{\lambda\alpha\rho}(\mu\alpha\rho)\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma s')t\Big)$.

(μ-E$^+$) If $r \twoheadrightarrow r'$ and $s \twoheadrightarrow s'$, then $r\mathrm{E}^+_\mu s \twoheadrightarrow r'\mathrm{E}^+_\mu s'$.

(β^+_μ) If $r \twoheadrightarrow C_{\mu\alpha\rho}t$ and $s \twoheadrightarrow s'$, then
$r\mathrm{E}^+_\mu\sigma s \twoheadrightarrow s'\Big(\mathsf{map}_{\lambda\alpha\rho}(\mu\alpha\rho)(\mu\alpha\rho\times\sigma)(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma s')x\rangle t\Big)$.

(The proofs are omitted.)
Define the reducing eliminations $\hat{\mathrm{E}}_\mu$ and $\hat{\mathrm{E}}^+_\mu$ by

$$r^{\mu\alpha\rho}\hat{\mathrm{E}}_\mu\sigma s \;:=\; \begin{cases} s\Big(\mathsf{map}_{\lambda\alpha\rho}(\mu\alpha\rho)\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma s)t\Big) & \text{, if } r = C_{\mu\alpha\rho}t \\ r\mathrm{E}_\mu\sigma s & \text{, else} \end{cases}$$

$$r^{\mu\alpha\rho}\hat{\mathrm{E}}^+_\mu\sigma s \;:=\; \begin{cases} s\Big(\mathsf{map}_{\lambda\alpha\rho}(\mu\alpha\rho)(\mu\alpha\rho\times\sigma)(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma s)x\rangle t\Big) & \text{, if } r = C_{\mu\alpha\rho}t \\ r\mathrm{E}^+_\mu\sigma s & \text{, else} \end{cases}$$

Unlike the situation of $\hat{\mathrm{E}}_\to$, $\hat{\mathrm{E}}_\forall$ and $\hat{\mathrm{E}}_\times$, these reducing eliminations never preserve normal forms if the first case applies. Therefore one may doubt whether the complete superdevelopment to be defined next deserves its name[3].

[2] This is the idea of completion: add rules (justified by the equality theory) until confluence is achieved.
[3] At least it deserves a name indicating that more is done than usual developments.

Extend the definition of the complete superdevelopment r^* of a term r as a term of the same type as r by recursion on terms and simultaneously prove that $r \twoheadrightarrow r^*$ as follows:

(0-E) $(r\mathrm{E}_0\rho)^* := r^*\mathrm{E}_0\rho$.

(1-I) $\mathsf{IN1}^* := \mathsf{IN1}$.

($\times$-I) $\langle r, s\rangle^* := \langle r^*, s^*\rangle$.

($\times$-E) $(r\mathsf{L})^* := r^*\hat{\mathrm{E}}_\times\mathsf{L}$ and $(r\mathsf{R})^* := r^*\hat{\mathrm{E}}_\times\mathsf{R}$ (see section 2.2.4 for the notation $\hat{\mathrm{E}}_\times$).

(+-I) $(\mathsf{INL}_\sigma r)^* := \mathsf{INL}_\sigma r^*$ and $(\mathsf{INR}_\rho r)^* := \mathsf{INR}_\rho r^*$.

(+-E) $(r\mathrm{E}_+ st)^* := r^*\hat{\mathrm{E}}_+ s^*t^*$.

(μ-I) $(C_{\mu\alpha\rho}t)^* := C_{\mu\alpha\rho}t^*$.

(μ-E) $(r\mathrm{E}_\mu\sigma s)^* := r^*\hat{\mathrm{E}}_\mu\sigma s^*$.

(μ-E$^+$) $(r\mathrm{E}^+_\mu\sigma s)^* := r^*\hat{\mathrm{E}}^+_\mu\sigma s^*$.

(The proofs are again omitted.)
We see that stratification does not enter this definition because in the definitions of $\hat{\mathrm{E}}_\mu$ and $\hat{\mathrm{E}}^+_\mu$ the term $\mathsf{map}_{\lambda\alpha\rho}$ is not completely super-developped, i. e., we do not use $\mathsf{map}^*_{\lambda\alpha\rho}$[4].
Now we check that the confluence proof for system F goes through with these definitions.
Reflexivity of $\twoheadrightarrow$ (Lemma 8.1) and its corollaries are immediate. Lemma 8.4 which essentially shows that $\twoheadrightarrow$ is parallel reduction goes through because in the cases (β_μ) and (β^+_μ) we know that $\mathsf{map}_{\lambda\alpha\rho}$ is closed. Compatibility with type substitution (Lemma 8.5) works in these critical cases because of the uniformity condition of ESMIT which says that $\mathsf{map}_{\lambda\alpha\rho}[\vec{\alpha} := \vec{\sigma}] = \mathsf{map}_{(\lambda\alpha\rho)[\vec{\alpha}:=\vec{\sigma}]}$.
Lemma 8.6 on parallel reducts of introduction terms is extended by the following

Lemma 8.11 (1-I) If $\mathsf{IN1} \twoheadrightarrow t$, then $t = \mathsf{IN1}$.

($\times$-I) If $\langle r, s\rangle \twoheadrightarrow t$, then $t = \langle r', s'\rangle$ with $r \twoheadrightarrow r'$ and $s \twoheadrightarrow s'$.

(+-I) If $\mathsf{INL}_\sigma r \twoheadrightarrow t$, then $t = \mathsf{INL}_\sigma r'$ with $r \twoheadrightarrow r'$. If $\mathsf{INR}_\rho r \twoheadrightarrow t$, then $t = \mathsf{INR}_\rho r'$ with $r \twoheadrightarrow r'$.

(μ-I) If $C_{\mu\alpha\rho}r \twoheadrightarrow t$, then $t = C_{\mu\alpha\rho}r'$ with $r \twoheadrightarrow r'$.

Proof As before by induction on $\twoheadrightarrow$ without using the induction hypothesis (also called inversion).□

The Maximality Lemma (Lemma 8.7) is the only part deserving some attention. The reasoning used in the proofs of its corollaries (including confluence) works in any abstract reduction system (= binary relation) and hence does not need any adjustment to ESMIT.

Lemma 8.12 (Maximality of r^* for ESMIT) If $r \twoheadrightarrow r'$, then $r' \twoheadrightarrow r^*$.

Proof Induction on $r \twoheadrightarrow r'$ as before. Only the cases (μ-I), (μ-E) and (β_μ) are shown.
(μ-I) Case $C_{\mu\alpha\rho}t \twoheadrightarrow C_{\mu\alpha\rho}t'$. By induction hypothesis $t' \twoheadrightarrow t^*$. Hence also $C_{\mu\alpha\rho}t' \twoheadrightarrow C_{\mu\alpha\rho}t^* = (C_{\mu\alpha\rho}t)^*$.
(μ-E) Case $r\mathrm{E}_\mu\sigma s \twoheadrightarrow r'\mathrm{E}_\mu\sigma s'$. By induction hypothesis $r' \twoheadrightarrow r^*$ and $s' \twoheadrightarrow s^*$. If $r^* = C_{\mu\alpha\rho}t$, then

[4] In PIT this could not even change anything because due to Lemma 5.2 $\mathsf{map}_{\lambda\alpha\rho}$ is already in normal form.

$r'\mathrm{E}_\mu\sigma s' \twoheadrightarrow s^*\Big(\mathsf{map}_{\lambda\alpha\rho}(\mu\alpha\rho)\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma s^*)t\Big) = (r\mathrm{E}_\mu\sigma s)^*$. Otherwise $r'\mathrm{E}_\mu\sigma s' \twoheadrightarrow r^*\mathrm{E}_\mu\sigma s^* = (r\mathrm{E}_\mu\sigma s)^*$.

(β_μ) Case $r\mathrm{E}_\mu\sigma s \twoheadrightarrow s'\Big(\mathsf{map}_{\lambda\alpha\rho}(\mu\alpha\rho)\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma s')t\Big)$. By induction hypothesis we have $C_{\mu\alpha\rho}t \twoheadrightarrow r^*$ and $s' \twoheadrightarrow s^*$. By the preceding lemma $r^* = C_{\mu\alpha\rho}t'$ with $t \twoheadrightarrow t'$. Hence $(r\mathrm{E}_\mu\sigma s)^* = s^*\Big(\mathsf{map}_{\lambda\alpha\rho}(\mu\alpha\rho)\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma s^*)t'\Big)$. By using reflexivity and the definition of $\twoheadrightarrow$ several times we get $s'\Big(\mathsf{map}_{\lambda\alpha\rho}(\mu\alpha\rho)\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma s')t\Big) \twoheadrightarrow s^*\Big(\mathsf{map}_{\lambda\alpha\rho}(\mu\alpha\rho)\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma s^*)t'\Big)$. $\square$

This establishes confluence for every system ESMIT (among them PIT, PITrec, SPIT, NIPIT and coMePIT) and the reader will hopefully be convinced that the other systems may be handled by the same apparatus. (Note that stratification did not enter neither the definitions nor the proofs. Hence, also ISMIT is covered. The need for closed $\mathsf{map}_{\lambda\alpha\rho}$ satisfying uniformity does not carry over to analogous conditions on the monotonicity witnesses of systems of monotone inductive types simply because they are "carried around" which makes the induction hypotheses in all the proofs apply. Hence, the situation becomes even easier for EMIT and its relatives. The systems in the spirit of Mendler are essentially nothing more than systems of non-interleaved positive inductive types and hence also do not pose new problems.)
There are other reasons why not so much effort is put into the confluence proofs: Because all of the systems are strongly normalizing (as will be shown in the next chapter), confluence follows from local confluence which is the content of Newman's Lemma (see e.g. [Mit96, p. 227]). And local confluence could be checked by a boring analysis of both reducts of the term in question (by double induction on the definition of $\to$ and consideration of every possible pair of rules including the congruence rules). An even stronger reason comes from the theory of higher-order rewrite systems: All of the rewrite rules in our systems are left-linear and non-overlapping. Those systems are called orthogonal[5] and are confluent in the first-order case. In [MN98] this result is lifted to Orthogonal Pattern Rewrite Systems which are higher-order rewrite systems using simply-typed lambda calculus as meta-language. The proof of this fact hopefully is extensible to cover our situation: One would have to extend the substitution calculus from simply-typed lambda calculus to system F with η-expansion which is also strongly normalizing and confluent (see [Joa97]) and therefore should allow to define the notions of [MN98] accordingly.

Note again that the induction principles expressed by the μ-types did not enter the proof of confluence. In the next chapter this will be different.

[5]If we added η-rules trivial critical pairs would arise and hence make the systems "weakly orthogonal".

Chapter 9

Strong normalization

Every system having a name in this thesis is strongly normalizing, i. e., $\mathsf{sn} = \Lambda$ holds for all of them. The proof might be carried out by proving strong normalization for the system NIPIT of non-interleaving positive inductive types or for the system $\mathsf{NISPIT{+}ex}$ of non-interleaving strictly positive inductive types including the existential quantifier and then referring to the reduction-preserving embeddings. I decided to prove strong normalization directly for some of the systems because this gives a lot of insight into the flexibility of the proof method by saturated sets which is a variant of Girard's candidate method. Two different proofs are given for every system: A proof where the definition of the computability predicate essentially fixes which introduction terms enter the predicate (the introduction-based proof[1]) and a proof where the definition of the computability predicate specifies that a term only may enter if sufficiently many eliminations with this term as main premise have the right properties (the elimination-based proof).
The informal modified realizability interpretation which served as a motivation in the earlier chapters now will lead to additional information after the proofs have been carried out: By considering the extracted type of the definition of the computability predicates encodings of the system at hand into other systems may be read off. All known examples of extracted encodings even preserve reduction (i. e., they are embeddings). An interesting project would be to justify this observation by a meta-theorem on proofs of strong normalization via the methods described in this chapter. It would be a refinement of the usual soundness theorem of modified realizability (see e. g. the presentations in [Ber93, vdP96b]).

9.1 Strong normalization of system F

> Anyone interested in mastering polymorphic lambda calculus should try to come up with one's own proof of normalization and see oneself make a mistake in the process. Then try giving a *correct* proof. Andre Scedrov [Sce89]

9.1.1 Terms in SN are strongly normalizing

As was promised in section 2.1.4 we prove that $\mathsf{SN} \subseteq \mathsf{sn}$. Note that we already know that $\mathsf{SN} \subseteq \mathsf{wn}$. If someone is only interested in weak normalization, he or she should should skip this section and proceed with section 9.1.2.

Lemma 9.1 If $x^\rho\vec{s}$ is a term and the terms in $\vec{s}$ are in sn, then $x^\rho\vec{s} \in \mathsf{sn}$.

Proof Main induction on the first term in $\vec{s}$ being in sn, side induction on the second being in sn, etc. Let r be any term such that $x^\rho\vec{s} \to r$. We have to show that $r \in \mathsf{sn}$. It is easy to see

[1] In [Pra71] (proof) terms in computability predicates of this kind are called strongly valid.

that $r = x^\rho\vec{s}\,'$ where $\vec{s}\,'$ is derived from $\vec{s}$ by reduction of one term. (This will henceforth be written as $\vec{s} \to \vec{s}\,'$.) Therefore the induction hypothesis applies to $x^\rho\vec{s}\,'$. □

Lemma 9.2 If $r \in \mathsf{sn}$, then $\lambda x^\rho r \in \mathsf{sn}$.

Proof Induction on $r \in \mathsf{sn}$. Let s be any term such that $\lambda x^\rho r \to s$. We have to show that $s \in \mathsf{sn}$. Obviously, $s = \lambda x^\rho r'$ with a term r' such that $r \to r'$. By induction hypothesis, $\lambda x^\rho r' \in \mathsf{sn}$. □

Lemma 9.3 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 $(\lambda x^\rho r)s\vec{s} \in \mathsf{sn}$.

Proof For ease of understanding the proof will first be given in a somewhat intuitive form: Main induction on $s \in \mathsf{sn}$, side induction[2] on $r[x^\rho := s]\vec{s} \in \mathsf{sn}$. Let t be any term such that $(\lambda x^\rho r)s\vec{s} \to t$. We have to show that $t \in \mathsf{sn}$.

- If $t = (\lambda x^\rho r')s\vec{s}$ with $r \to r'$, then $r[x^\rho := s]\vec{s} \to r'[x^\rho := s]\vec{s}$. By side induction hypothesis, $t \in \mathsf{sn}$.
- If $t = (\lambda x^\rho r)s'\vec{s}$ with $s \to s'$, then $r[x^\rho := s]\vec{s} \to^* r[x^\rho := s']\vec{s}$, hence also $r[x^\rho := s']\vec{s} \in \mathsf{sn}$. By main induction hypothesis, $t \in \mathsf{sn}$.
- If $t = (\lambda x^\rho r)s\vec{s}\,'$ with $\vec{s} \to \vec{s}\,'$, then $r[x^\rho := s]\vec{s} \to r[x^\rho := s]\vec{s}\,'$. By side induction hypothesis, $t \in \mathsf{sn}$.
- If $t = r[x^\rho := s]\vec{s}$, then $t \in \mathsf{sn}$ by assumption.
- There are no other possibilities (which can easily shown by induction on the inductive definition of $\to$).

As this lemma is crucial to the proof of strong normalization and to my knowledge is an unavoidable[3] part of any strong normalization proof by the Tait method, I give a second proof which should be seen as justification for the preceding proof. The very first proof of strong normalization (in [Tai75]) had a similar lemma which was proved by refuting the existence of infinite reduction sequences. My aim is to avoid reasoning by reduction sequences and to confine to the use of the induction principle coming from the inductive definition of the set sn.
Fix a variable x^ρ and the length of all vectors (and the types of the terms in the vectors) in the proof to follow. Define

$$M := \{s | \forall r, \vec{s} : r[x^\rho := s]\vec{s} \in \mathsf{sn} \Rightarrow (\lambda x^\rho r)s\vec{s} \in \mathsf{sn}\}.$$

We have to show that $\mathsf{sn} \subseteq M$. We do induction on sn. Let s be a term. We assume that for every term s' such that $s \to s'$ we have $s' \in M$. We have to show that $s \in M$. Assume $r[x^\rho := s]\vec{s} \in \mathsf{sn}$. We have to show $(\lambda x^\rho r)s\vec{s} \in \mathsf{sn}$. Define

$$N := \{t \in \mathsf{sn} | \forall r, \vec{s} : t = r[x^\rho := s]\vec{s} \Rightarrow (\lambda x^\rho r)s\vec{s} \in \mathsf{sn}\}.$$

[2] Note that the term $r[x^\rho := s]\vec{s}$ of the side induction depends on the term s of the main induction.

[3] In [Gal98] it is argued that the proof presented there does not "involve rather strange looking terms such as $M[N/x]N_1 \dots N_k$" (p. 233). On p. 236 it is conceded that "it is no surprise that they crop up again" but that they do not have to be dealt with explicitly. On p. 250, it becomes clear that those terms in fact do not enter the refined arrangement of the proof. But there is a reference to finite reduction sequences in property (P3) on p. 248 which is the price to pay which to me seems to be higher than to consider $r[x^\rho := s]\vec{s}$. Therefore I stick to the word "unavoidable" but do not take it too serious.

It suffices to show that $\mathsf{sn} \subseteq N$. We do induction[4] on sn. Let t be a term. We assume that for every term t' such that $t \to t'$ we have $t' \in N$. We have to show that $t \in N$. $t \in \mathsf{sn}$ follows[5] from $N \subseteq \mathsf{sn}$. Let $t = r[x^\rho := s]\vec{s}$ for some r and $\vec{s}$. We have to show $(\lambda x^\rho r)s\vec{s} \in \mathsf{sn}$. Let v be any term such that $(\lambda x^\rho r)s\vec{s} \to v$ and show $v \in \mathsf{sn}$.

- If $v = (\lambda x^\rho r')s\vec{s}$ with $r \to r'$, then $r[x^\rho := s]\vec{s} \to r'[x^\rho := s]\vec{s}$, hence $r'[x^\rho := s]\vec{s} \in N$. Therefore $v \in \mathsf{sn}$.
- If $v = (\lambda x^\rho r)s'\vec{s}$ with $s \to s'$, then $r[x^\rho := s]\vec{s} \to^* r[x^\rho := s']\vec{s}$, hence also $r[x^\rho := s']\vec{s} \in \mathsf{sn}$. Because of $s' \in M$, also $v \in \mathsf{sn}$.
- If $v = (\lambda x^\rho r)s\vec{s}'$ with $\vec{s} \to \vec{s}'$, then $r[x^\rho := s]\vec{s} \to r[x^\rho := s]\vec{s}'$, hence $r[x^\rho := s]\vec{s}' \in N$. Therefore $v \in \mathsf{sn}$.
- If $v = r[x^\rho := s]\vec{s}$, then $v = t \in \mathsf{sn}$.
- There are no other possibilities (to be shown as in the first proof).

We conclude that the second proof is nothing but the first proof with explicit statements of the induction formulas added. □

Lemma 9.4 If $\Lambda\alpha r$ is a term and $r \in \mathsf{sn}$, then $\Lambda\alpha r \in \mathsf{sn}$.

Proof Induction on $r \in \mathsf{sn}$. Let s be any term such that $\Lambda\alpha r \to s$. We have to show that $s \in \mathsf{sn}$. Obviously, $s = \Lambda\alpha r'$ with a term r' such that $r \to r'$. By induction hypothesis, $\Lambda\alpha r' \in \mathsf{sn}$. □

Lemma 9.5 If $(\Lambda\alpha r)\sigma\vec{s}$ is a term and $r[\alpha := \sigma]\vec{s} \in \mathsf{sn}$, then $(\Lambda\alpha r)\sigma\vec{s} \in \mathsf{sn}$.

Proof Induction on $r[\alpha := \sigma]\vec{s} \in \mathsf{sn}$. Let t be any term such that $(\Lambda\alpha r)\sigma\vec{s} \to t$. We have to show that $t \in \mathsf{sn}$.

- If $t = (\Lambda\alpha r')\sigma\vec{s}$ with $r \to r'$, then $r[\alpha := \sigma]\vec{s} \to r'[\alpha := \sigma]\vec{s}$, hence $t \in \mathsf{sn}$ by induction hypothesis.
- If $t = (\Lambda\alpha r)\sigma\vec{s}'$ with $\vec{s} \to \vec{s}'$, then $r[\alpha := \sigma]\vec{s} \to r[\alpha := \sigma]\vec{s}'$, hence $t \in \mathsf{sn}$ by induction hypothesis.
- If $t = r[\alpha := \sigma]\vec{s}$, then $t \in \mathsf{sn}$ by assumption.
- There are no other possibilities.

Note that this proof is essentially easier than the one covering $(\beta_\to)$. □

Lemma 9.6 $\mathsf{SN} \subseteq \mathsf{sn}$.

Proof The preceding lemmas simply say that the defining properties of SN are closure properties of sn. □

We remark that types do not play a role in the proof and that therefore $\mathsf{SN} \subseteq \mathsf{sn}$ could also be proved for a untyped term system.

[4]This is the same as proving $\mathsf{sn} \subseteq N' := \{t \text{ term} | \forall r, \vec{s} : t = r[x^\rho := s]\vec{s} \Rightarrow (\lambda x^\rho r)s\vec{s} \in \mathsf{sn}\}$ by extended induction (as defined in the introduction to chapter 4) on sn.

[5]The proof by the principle of extended induction would not need this (in this case very short) reasoning because it is once done in the derivation of that principle from the induction principle.

9.1.2 Saturated sets

The main goal of this chapter is to establish that every term is in SN which of course is not true for untyped terms. The following proof is by the Tait method of computability predicates. In its very first form it appeared in [Tai67]. Due to the presence of type quantification the notion of computability predicate has to be relativized to any sensible predicate which Girard called "candidats de reductibilité" [Gir72] (see [Gal90] for a discussion of several variations on the definition). The least criterion for being sensible is that only strongly normalizing terms may enter the candidates. Girard's candidates in [GLT89] are closed under reduction and "neutral" terms (i. e., terms which are no abstractions) enter if all their reducts are already in the set. Tait's candidates ("propositions" in [Tai75]) are also closed under reduction but this is never used in his proof of strong normalization. His essential condition for $\mathcal{M}$ being a candidate is

$$r[x := s]\vec{s} \in \mathcal{M} \wedge \ s \text{ is strongly normalizing } \Rightarrow (\lambda x r)s\vec{s} \in \mathcal{M}.$$

Thorsten Altenkirch observed that this is a pattern which may be generalized considerably: One may base the candidates on weak head expansion[6] under the proviso that this expansion does not lead out of the set of strongly normalizing term. Those sets are then called saturated (a name which is also used in [Bar93] for candidates in the spirit of [Tai75]). In [Alt93a] this approach is used for the calculus of constructions extended by some generalization of trees. In the following proof the explicit reference to strongly normalizing terms is abandoned in favour of the set SN.[7] The definition of saturated sets is modelled after the definition of SN. In this thesis only typed terms are considered. Therefore also the saturated sets consist only of typed terms as opposed e. g. to the treatment in [Alt93a]. In my view this makes the definition more intuitive: All the terms in a saturated set are required to have the same type, and saturated sets by definition have all the closure properties of SN which do not conflict with this requirement. It is possible to show that by adding closure under reduction one gets back[8] Girard's candidates as in [GLT89].

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 $x^\rho\vec{s} \in \mathcal{M}$.

$(\beta_\to)$ 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 $(\Lambda\alpha r)\sigma\vec{s} \in \mathcal{M}$.

Note that the rules ($\to$-I) and ($\forall$-I) of the definition of SN do not enter this definition because they change the type of the term.

Let SAT_τ be the set of all τ-saturated sets and $\mathsf{SAT} := \bigcup_\tau \mathsf{SAT}_\tau$. Elements of SAT are called saturated sets. Obviously, $\mathsf{SN}_\tau \in \mathsf{SAT}_\tau$. Note that SAT_τ is closed under arbitrary intersections

[6]In [MKO95] this process is reversed in some sense: The computability predicates have a very easy definition but instead of checking whether a term is in the predicate one checks whether its weak head normal form is in the predicate. The price to pay is that in some sense one has to calculate with equivalence classes modulo weak head reduction.

[7]At first sight, the approach taken in [BTG91] of studying arbitrary "candidate-closed" sets instead of the strongly normalizing terms, seems to be even more general. However, it turns out that any candidate-closed set is a superset of SN and hence both approaches are equivalent (ignoring algebraic term rewriting which is also present in [BTG91]).

[8]Hence, Girard's candidates are nothing but saturated sets which are closed under reduction. I consider the freedom from analyses of the possible reducts (except when showing that $\mathsf{SN} \subseteq \mathsf{sn}$ where it conceptually cannot be avoided) as the main advantage of the method shown in this thesis.

(the empty intersection being SN_τ) and hence is a complete lattice. For $\mathcal{M} \in \mathsf{SAT}$ write $\mathcal{M}^\tau$ to indicate that $\mathcal{M} \in \mathsf{SAT}_\tau$.
If one is only interested in weak normalization, one could leave out the condition $s \in \mathsf{SN}$ from $(\beta_\to)$ which would clearly amount to basing the definition of saturatedness on WN instead of SN. Then one would have $\mathsf{WN} \cap \Lambda_\tau \in \mathsf{SAT}_\tau$ and would finally get that $\mathsf{WN} = \Lambda$. But this does not simplify any of the arguments to follow. And of course, there would be no justification of an embedding extracted from the proof of normalization (see section 9.3).
Let M be a set of terms. The τ-saturated closure $\mathsf{cl}_\tau(M)$ of M is defined inductively by the following clauses.

($\subseteq$) If $r \in M \cap \mathsf{SN}_\tau$, then $r \in \mathsf{cl}_\tau(M)$.

(V) If $x^\rho \vec{s}$ is a term of type τ and the terms in $\vec{s}$ are in SN, then $x^\rho \vec{s} \in \mathsf{cl}_\tau(M)$.

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

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

This is a strictly positive inductive definition. By induction on the definition one shows $\mathsf{cl}_\tau(M) \subseteq \mathsf{SN}_\tau$. Therefore, $\mathsf{cl}_\tau(M)$ is the smallest τ-saturated set containing $M \cap \mathsf{SN}_\tau$. Obviously, cl_τ is an isotone mapping of sets of terms to SAT_τ.

Calculating with saturated sets

We define operations on saturated sets which correspond to the type constructions.

Arrow types We define an arrow construction for saturated sets, i. e., given a ρ-saturated set $\mathcal{M}$ and a σ-saturated set $\mathcal{N}$, we define a $(\rho \to \sigma)$-saturated set $\mathcal{M} \to \mathcal{N}$. There are mainly two ways how to do that: In the introduction-based definition, sufficiently many λ-abstractions are put into the set whereas in the elimination-based definition only terms enter the set which behave well with respect to $\to$-elimination. Define

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

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 $\mathcal{M}$.

Lemma 9.7 $M_E \cap \mathsf{SN} \in \mathsf{SAT}$ (which implies $\mathcal{M} \to_E \mathcal{N} = M_E \cap \mathsf{SN}$).

Proof Very easy. One only has to append the term s to the vector $\vec{s}$. □

Note that we do not immediately see that $M_E \subseteq \mathsf{SN}$. Some more traditional proofs of strong normalization would require to prove it (where SN would be replaced by sn), and it would be done in the following way: Let x^ρ be a variable. Hence $x^\rho \in \mathcal{M}$. By assumption $rx \in \mathcal{N}$, hence $rx \in \mathsf{SN}$. Because r is a subterm of rx, also $r \in \mathsf{SN}$. A similar reasoning is indeed possible with our set SN, but only for $x^\rho \notin \mathsf{FV}(r)$ which does no harm. But in our approach we do not need the underlying lemma, viz. $rx^\rho \in \mathsf{SN} \wedge x^\rho \notin \mathsf{FV}(r) \Rightarrow r \in \mathsf{SN}$ which in some sense contradicts the philosophy of only using the reduction strategy expressed in the definition of SN. (Ex post—after the proof of the main theorem which is $\mathsf{SN} = \Lambda$—we know that it is true in a trivial sense.)

Lemma 9.8 $M_I \subseteq \mathsf{SN}$ (which implies $\mathcal{M} \to_I \mathcal{N} \supseteq M_I$).

Proof Choose $s := x^\rho$, which trivially is in $\mathcal{M}$. Then $t \in \mathcal{N}$ by definition, hence $t \in \mathsf{SN}$ and consequently $r = \lambda x^\rho t \in \mathsf{SN}$. □

Lemma 9.9 $\mathcal{M} \to_I \mathcal{N} \subseteq \mathcal{M} \to_E \mathcal{N}$.

Proof It suffices to show $M_I \subseteq M_E$. This is an application of the rule $(\beta_\to)$ of the definition of saturated sets (with empty $\vec{s}$). □

The preceding three statements immediately imply the following rules for calculating with $\mathcal{M} \to_I \mathcal{N}$ and $\mathcal{M} \to_E \mathcal{N}$:

($\to_E$-E) If $r \in \mathcal{M} \to_E \mathcal{N}$ and $s \in \mathcal{M}$, then $r\mathrm{E}_\to s \in \mathcal{N}$.

($\to_I$-E) If $r \in \mathcal{M} \to_I \mathcal{N}$ and $s \in \mathcal{M}$, then $r\mathrm{E}_\to s \in \mathcal{N}$.

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

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

In the sequel $\mathcal{M} \to \mathcal{N}$ will denote either $\mathcal{M} \to_I \mathcal{N}$ or $\mathcal{M} \to_E \mathcal{N}$ as long as it does not matter which one is meant, i.e. the use of this set is restricted to the following rules[9]:

(SAT) $\mathcal{M} \to \mathcal{N} \in \mathsf{SAT}_{\rho\to\sigma}$.

($\to$-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 $r\mathrm{E}_\to s \in \mathcal{N}$.

It is possible that $\mathcal{M} \to_I \mathcal{N} \neq \mathcal{M} \to_E \mathcal{N}$: Let $\rho, \sigma := \alpha \to \alpha$, $\mathcal{M} := \mathsf{SN}_{\alpha\to\alpha}$ and $\mathcal{N} := \mathsf{cl}_{\alpha\to\alpha}(N)$ with $N := \{(\lambda z^{\alpha\to\alpha} z)t | t \in \mathsf{SN}_{\alpha\to\alpha}\}$. Clearly, $N \subseteq \mathsf{SN}_{\alpha\to\alpha}$ and hence $N \subseteq \mathcal{N}$.
Let $r := \lambda z^{\alpha\to\alpha} z$. We claim that $r \in \mathcal{M} \to_E \mathcal{N} \setminus \mathcal{M} \to_I \mathcal{N}$.
We want to show $r \in \mathcal{M} \to_E \mathcal{N}$. $r \in \mathsf{SN}_{(\alpha\to\alpha)\to\alpha\to\alpha}$. Hence it suffices to show $rt \in \mathcal{N}$ for every $t \in \mathsf{SN}_{\alpha\to\alpha}$. By definition $rt \in N$ and therefore also $rt \in \mathcal{N}$.
For every set M, we have that $\lambda x^\rho r \in \mathsf{cl}(M) \Rightarrow \lambda x^\rho r \in M$. This is easily seen by induction on the definition of the saturated closure cl.
Assume that $r \in \mathcal{M} \to_I \mathcal{N}$. By the preceding observation, $r \in M_I$. Hence for every $t \in \mathsf{SN}_{\alpha\to\alpha}$, $t \in \mathcal{N}$. Take $t := \lambda x^\alpha x$. Therefore $\lambda x^\alpha x \in \mathcal{N}$ and again by the above observation $\lambda x^\alpha x \in N$ which is not the case.

[9]This would not suffice to get the normalization proof through in extensions of the system having introduction constants of arrow type instead of term forming rules (e.g. injection constants for sum types instead of INL_ρ and INR_ρ in eF which are not terms but term formers). In those systems one would have to base the definitions of computability predicates on $\mathcal{M} \to_E \mathcal{N}$ and thus would be forced to follow an elimination-based approach concerning the arrow type. More disturbing on the aesthetic side would be the obligation to unwind the definition of $\mathcal{M} \to_E \mathcal{N}$ several times in the proof instead of only using the above rules which directly correspond to a semantical view on the introduction and elimination rule for terms. This aesthetic defect will affect future formalizations of the arguments and hence the process of extraction of embeddings from normalization proofs.

Universal types 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

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

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.

Lemma 9.10 $M_E \cap \mathsf{SN} \in \mathsf{SAT}$ (which implies $\forall_E(\Phi) = M_E \cap \mathsf{SN}$).

Proof As easy as for the arrow type (simply append τ to $\vec{s}$). □

Lemma 9.11 $M_I \subseteq \mathsf{SN}$ (which implies $\forall_I(\Phi) \supseteq M_I$).

Proof Choose $\tau := \alpha$ and $\mathcal{P} := \mathsf{SN}_\alpha$. Hence, $t \in \Phi_\alpha(\mathsf{SN}_\alpha) \subseteq \mathsf{SN}$. Therefore $\Lambda\alpha t \in \mathsf{SN}$. □

Lemma 9.12 $\forall_I(\Phi) \subseteq \forall_E(\Phi)$.

Proof It suffices to show $M_I \subseteq M_E$. This is an application of the rule $(\beta_\forall)$ of the definition of saturated sets (with empty $\vec{s}$). □

As for arrow types we get the following rules:

($\forall_E$-E) If $r \in \forall_E(\Phi)$, then $\forall\tau\forall\mathcal{P} \in \mathsf{SAT}_\tau\ r\mathrm{E}_\forall\tau \in \Phi_\tau(\mathcal{P})$.

($\forall_I$-E) If $r \in \forall_I(\Phi)$, then $\forall\tau\forall\mathcal{P} \in \mathsf{SAT}_\tau\ r\mathrm{E}_\forall\tau \in \Phi_\tau(\mathcal{P})$.

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

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

(The introduction rules are under the proviso that $\Lambda\alpha t$ is a term.)
From now on $\forall(\Phi)$ shall denote $\forall_I(\Phi)$ or $\forall_E(\Phi)$ if only the following rules are used.

(SAT) $\forall(\Phi) \in \mathsf{SAT}_{\forall\alpha\rho}$.

($\forall$-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 $\forall\tau\forall\mathcal{P} \in \mathsf{SAT}_\tau\ r\mathrm{E}_\forall\tau \in \Phi_\tau(\mathcal{P})$.

It is possible that $\forall_I(\Phi) \neq \forall_E(\Phi)$. Let $\rho := \beta \to \beta$ with $\beta \neq \alpha$. Let $\mathcal{M} := \mathsf{cl}_{\beta\to\beta}(M)$ with $M := \{(\Lambda\alpha r)\tau | \tau \text{ type}, r \in \mathsf{SN}_{\beta\to\beta} \text{ with } \alpha \notin \mathsf{FTV}(r)\}$. Because $r[\alpha := \tau] = r$, $M \subseteq \mathsf{SN}_{\beta\to\beta}$ and hence $M \subseteq \mathcal{M} \in \mathsf{SAT}_{\beta\to\beta}$. Let $\Phi_\tau(\mathcal{P}) := \mathcal{M}$ for all τ and all $\mathcal{P} \in \mathsf{SAT}_\tau$.
Let $r := \Lambda\alpha\lambda x^\beta x$. We claim that $r \in \forall_E(\Phi) \setminus \forall_I(\Phi)$.
We want to show $r \in \forall_E(\Phi)$. $r \in \mathsf{SN}_{\forall\alpha.\beta\to\beta}$. Let τ be a type and $\mathcal{P} \in \mathsf{SAT}_\tau$. It suffices to show that $r\tau \in \Phi_\tau(\mathcal{P}) = \mathcal{M}$. By definition $r\tau \in M \subseteq \mathcal{M}$.
For every set M, we have that $\Lambda\alpha r \in \mathsf{cl}(M) \Rightarrow \Lambda\alpha r \in M$. This is easily seen by induction on the definition of the saturated closure cl.
Assume that $r \in \forall_I(\Phi)$. By the preceding observation, $r \in M_I$. Hence for every type τ and every $\mathcal{P} \in \mathsf{SAT}_\tau$: $(\lambda x^\beta x)[\alpha := \tau] \in \Phi_\tau(\mathcal{P})$, whence $\lambda x^\beta x \in \mathcal{M}$. By the observation on cl at the end of the section on arrow types we conclude $\lambda x^\beta x \in M$ which is not the case.

Computability predicates

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 as follows.

(V) $\mathsf{SC}_\alpha[\vec{\alpha} := \vec{\mathcal{P}}] := \mathsf{cl}_\alpha(\emptyset)$[10], if α does not occur in the list $\vec{\alpha}$.
$\mathsf{SC}_\alpha[\vec{\alpha} := \vec{\mathcal{P}}] := \mathcal{P}_i$, if i is the smallest index such that $\alpha_i = \alpha$.

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

($\forall$) $\mathsf{SC}_{\forall\alpha\rho}[\vec{\alpha} := \vec{\mathcal{P}}] := \forall(\Phi)$ 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 $\alpha \notin \vec{\alpha} \cup \mathsf{FV}(\vec{\sigma})$ and use Lemma 2.6 to conclude that this definition is type-correct).

The definition in case ($\forall$) may be written more intuitively as follows:

$$\mathsf{SC}_{\forall\alpha\rho}[\vec{\alpha} := \vec{\mathcal{P}}] := \forall\mathcal{P}.\mathsf{SC}_\rho[\vec{\alpha}, \alpha := \vec{\mathcal{P}}, \mathcal{P}]$$

The implicit quantification over all types σ hopefully will always be clear from the context. This notation will be used in the sequel (especially in the normalization proofs for the systems with inductive types).
Obviously, the indetermination of whether the introduction-based or the elimination-based constructions for saturated sets are used has to be handled with care. E. g. the choice for $\to_I$ or $\to_E$ in the definition of $\mathsf{SC}_{\rho\to\sigma}[\vec{\alpha} := \vec{\mathcal{P}}]$ must not depend on $\mathcal{P}_i$ with $\alpha_i \notin \mathsf{FV}(\rho \to \sigma)$ because otherwise the following lemma would fail to hold. The next but one lemma imposes much stronger restrictions: The choice has to be the same for any substitution instance of $\rho \to \sigma$. To be on the safe side, simply assume that e. g. $\to$ means $\to_E$ **everywhere** and $\forall(\Phi)$ means $\forall_I(\Phi)$ **everywhere**. Hence, the freedom in the choice reduces to 4 possibilities. And the computability predicates indeed depend on this choice: The examples that $\mathcal{M} \to_I \mathcal{N} \neq \mathcal{M} \to_E \mathcal{N}$ and $\forall_I(\Phi) \neq \forall_E(\Phi)$ suffice because $\mathsf{SC}_{\beta\to\gamma}[\beta, \gamma := \mathcal{M}, \mathcal{N}] = \mathcal{M} \to \mathcal{N}$ (generally for $\beta \neq \gamma$) and (for $\gamma \notin \{\alpha, \beta\}$) $\mathsf{SC}_{\forall\alpha\gamma}[\gamma := \mathcal{M}] = \forall(\Phi)$ with $\Phi_\tau(\mathcal{P}) = \mathsf{SC}_\gamma[\gamma, \alpha := \mathcal{M}, \mathcal{P}] = \mathcal{M}$ (for the specific example shown on p. 117).

Lemma 9.13 If $\alpha \notin \mathsf{FV}(\rho)$, then $\mathsf{SC}_\rho[\vec{\alpha}, \alpha := \vec{\mathcal{P}}, \mathcal{P}] = \mathsf{SC}_\rho[\vec{\alpha} := \vec{\mathcal{P}}]$.

Proof By induction on ρ show that even $\mathsf{SC}_\rho[\vec{\alpha}, \alpha, \vec{\alpha}' := \vec{\mathcal{P}}, \mathcal{P}, \vec{\mathcal{P}}'] = \mathsf{SC}_\rho[\vec{\alpha}, \vec{\alpha}' := \vec{\mathcal{P}}, \vec{\mathcal{P}}']$ holds if $\alpha \notin \mathsf{FV}(\rho)$. □

Lemma 9.14 If $\alpha \notin \vec{\alpha}$, then $\mathsf{SC}_{\rho[\alpha:=\tau]}[\vec{\alpha} := \vec{\mathcal{P}}] = \mathsf{SC}_\rho[\vec{\alpha}, \alpha := \vec{\mathcal{P}}, \mathsf{SC}_\tau[\vec{\alpha} := \vec{\mathcal{P}}]]$.

Proof By induction on ρ show that for $\alpha \notin \vec{\alpha} \cup \vec{\alpha}'$ we have

$$\mathsf{SC}_{\rho[\alpha:=\tau]}[\vec{\alpha}, \vec{\alpha}' := \vec{\mathcal{P}}, \vec{\mathcal{P}}'] = \mathsf{SC}_\rho[\vec{\alpha}, \alpha, \vec{\alpha}' := \vec{\mathcal{P}}, \mathsf{SC}_\tau[\vec{\alpha}, \vec{\alpha}' := \vec{\mathcal{P}}, \vec{\mathcal{P}}'], \vec{\mathcal{P}}'].$$

The preceding lemma is needed in the case ($\forall$). □

Corollary 9.15 If $\beta \notin \vec{\alpha} \cup \mathsf{FV}(\rho)$ and $\alpha \notin \vec{\alpha}$, then

$$\mathsf{SC}_{\rho[\alpha:=\beta]}[\vec{\alpha}, \beta := \vec{\mathcal{P}}, \mathcal{P}] = \mathsf{SC}_\rho[\vec{\alpha}, \alpha := \vec{\mathcal{P}}, \mathcal{P}].$$

[10] One could take any set instead of $\emptyset$.

Proof Trivial, if $\alpha = \beta$. Otherwise

$$\begin{aligned}\mathsf{SC}_{\rho[\alpha:=\beta]}[\vec{\alpha},\beta := \vec{\mathcal{P}},\mathcal{P}] &= \mathsf{SC}_\rho[\vec{\alpha},\beta,\alpha := \vec{\mathcal{P}},\mathcal{P},\mathsf{SC}_\beta[\vec{\alpha},\beta := \vec{\mathcal{P}},\mathcal{P}]] \quad \text{by the preceding lemma}\\ &= \mathsf{SC}_\rho[\vec{\alpha},\alpha := \vec{\mathcal{P}},\mathcal{P}] \quad \text{due to the proof of the last but one lemma.}\end{aligned}$$

□

Lemma 9.13 is the "coincidence lemma" and Lemma 9.14 is the "substitution lemma" which are auxiliary statements for the next lemma which should be called the "soundness lemma". This next lemma shows that every term is strongly computable—even if term variables are substituted by strongly computable terms. This generalization is needed in the case (→-I) of the proof. The case (∀-I) needs the freedom in the number of saturated sets to which the notion of strong computability is relativized. The proviso in the statement (see below) is only present for technical reasons: It is tailored to make the statement true for variables! This technical problem arises because of the natural deduction formulation of system F as opposed to a formulation with contexts which many authors prefer.

Lemma 9.16 Let r^ρ be a term, $\vec{x}^{\vec{\rho}}$ a list of variables and $\vec{s}$ an equally long list of terms such that $\vec{s} \in \mathsf{SC}_{\vec{\rho}}[\vec{\alpha} := \vec{\mathcal{P}}^{\vec{\sigma}}]$ for some $\vec{\mathcal{P}} \subseteq \mathsf{SAT}$. Assume that

$$\forall x \forall \pi \forall \pi'.(x^\pi, x^{\pi'} \in \mathsf{FV}(r) \cup \vec{x}^{\vec{\rho}}) \wedge (\pi[\vec{\alpha} := \vec{\sigma}] = \pi'[\vec{\alpha} := \vec{\sigma}]) \Rightarrow \pi = \pi'.$$

Then $r[\vec{\alpha} := \vec{\sigma}][\vec{x}^{\vec{\rho}[\vec{\alpha}:=\vec{\sigma}]} := \vec{s}] \in \mathsf{SC}_\rho[\vec{\alpha} := \vec{\mathcal{P}}]$.

Proof Induction on r. Abbreviate $r[\vec{\alpha} := \vec{\sigma}][\vec{x}^{\vec{\rho}[\vec{\alpha}:=\vec{\sigma}]} := \vec{s}]$ by r' (also for any other terms instead of r). The assumption in the statement will be referenced to as "injectivity".
(V) Case x^ρ. If $x^\rho \notin \vec{x}^{\vec{\rho}}$, then $x^{\rho[\vec{\alpha}:=\vec{\sigma}]} \notin \vec{x}^{\vec{\rho}[\vec{\alpha}:=\vec{\sigma}]}$. (For a proof assume $x^{\rho[\vec{\alpha}:=\vec{\sigma}]} = x_i^{\rho_i[\vec{\alpha}:=\vec{\sigma}]}$ for some i. Hence $x = x_i$ and $\rho[\vec{\alpha} := \vec{\sigma}] = \rho_i[\vec{\alpha} := \vec{\sigma}]$. Because $x^\rho, x_i^{\rho_i} \in \mathsf{FV}(x^\rho) \cup \vec{x}^{\vec{\rho}}$, this implies $\rho = \rho_i$, and consequently $x^\rho = x_i^{\rho_i} \in \vec{x}^{\vec{\rho}}$ which is not the case.)

$$(x^\rho)' = x^{\rho[\vec{\alpha}:=\vec{\sigma}]}[\vec{x}^{\vec{\rho}[\vec{\alpha}:=\vec{\sigma}]} := \vec{s}] = x^{\rho[\vec{\alpha}:=\vec{\sigma}]} \in \mathsf{SC}_\rho[\vec{\alpha} := \vec{\mathcal{P}}],$$

because $\mathsf{SC}_\rho[\vec{\alpha} := \vec{\mathcal{P}}] \in \mathsf{SAT}_{\rho[\vec{\alpha}:=\vec{\sigma}]}$.
If $x^\rho \in \vec{x}^{\vec{\rho}}$, then let i be the smallest number such that $x_i^{\rho_i[\vec{\alpha}:=\vec{\sigma}]} = x^{\rho[\vec{\alpha}:=\vec{\sigma}]}$. Due to injectivity, $\rho_i = \rho$. Hence

$$(x^\rho)' = x_i^{\rho_i[\vec{\alpha}:=\vec{\sigma}]}[\vec{x}^{\vec{\rho}[\vec{\alpha}:=\vec{\sigma}]} := \vec{s}] = s_i \in \mathsf{SC}_{\rho_i}[\vec{\alpha} := \vec{\mathcal{P}}] = \mathsf{SC}_\rho[\vec{\alpha} := \vec{\mathcal{P}}].$$

(→-I) Case $\lambda x^\rho r^\sigma$. We may assume (1): $x^\pi \in \mathsf{FV}(r) \Rightarrow \pi = \rho$. Therefore $(\lambda x^\rho r)[\vec{\alpha} := \vec{\sigma}] = \lambda x^{\rho[\vec{\alpha}:=\vec{\sigma}]}.r[\vec{\alpha} := \vec{\sigma}]$. We may assume (2): $x^{\rho[\vec{\alpha}:=\vec{\sigma}]} \notin \vec{x}^{\vec{\rho}[\vec{\alpha}:=\vec{\sigma}]} \cup \mathsf{FV}(\vec{s})$. Therefore $(\lambda x^\rho r)' = \lambda x^{\rho[\vec{\alpha}:=\vec{\sigma}]}.r'$. We have to show $\lambda x^{\rho[\vec{\alpha}:=\vec{\sigma}]}.r' \in \mathsf{SC}_{\rho\to\sigma}[\vec{\alpha} := \vec{\mathcal{P}}]$. Use (→-I) for saturated sets. Let $s \in \mathsf{SC}_\rho[\vec{\alpha} := \vec{\mathcal{P}}]$. Show that $r'[x^{\rho[\vec{\alpha}:=\vec{\sigma}]} := s] \in \mathsf{SC}_\sigma[\vec{\alpha} := \vec{\mathcal{P}}]$. Because of (2) and Lemma 2.14,

$$r'[x^{\rho[\vec{\alpha}:=\vec{\sigma}]} := s] = r[\vec{\alpha} := \vec{\sigma}][\vec{x}^{\vec{\rho}[\vec{\alpha}:=\vec{\sigma}]}, x^{\rho[\vec{\alpha}:=\vec{\sigma}]} := \vec{s}, s].$$

By induction hypothesis for r this term is in $\mathsf{SC}_\rho[\vec{\alpha} := \vec{\mathcal{P}}]$ provided injectivity holds for this situation. If $y^\pi, y^{\pi'} \in (\mathsf{FV}(r) \setminus \{x^\rho\}) \cup \vec{x}^{\vec{\rho}}$, then by assumption $\pi[\vec{\alpha} := \vec{\sigma}] = \pi'[\vec{\alpha} := \vec{\sigma}]$ implies $\pi = \pi'$. Hence we only have to consider x^ρ and $x^\pi \in \mathsf{FV}(r) \cup \vec{x}^{\vec{\rho}}$. If $x^\pi \in \mathsf{FV}(r)$, then $\rho = \pi$ by (1). If $x^\pi \in \vec{x}^{\vec{\rho}}$, then $x^\pi = x_i^{\rho_i}$ for some i. $\rho[\vec{\alpha} := \vec{\sigma}] = \rho_i[\vec{\alpha} := \vec{\sigma}]$ is impossible due to (2), hence trivially $\rho[\vec{\alpha} := \vec{\sigma}] = \pi[\vec{\alpha} := \vec{\sigma}] \Rightarrow \rho = \pi$.

(→-E) Case $r^{\rho\to\sigma}s^\rho$. $(rs)' = r's'$. Because $\mathsf{FV}(rs) = \mathsf{FV}(r) \cup \mathsf{FV}(s)$, injectivity holds also for r and s. By induction hypothesis $r' \in \mathsf{SC}_{\rho\to\sigma}[\vec{\alpha} := \vec{\mathcal{P}}]$ and $s' \in \mathsf{SC}_\rho[\vec{\alpha} := \vec{\mathcal{P}}]$, which implies $r's' \in \mathsf{SC}_\sigma[\vec{\alpha} := \vec{\mathcal{P}}]$ by (→-E) for saturated sets.
(∀-I) Case $\Lambda\alpha r^\rho$. We may assume that (1): $\alpha \notin \vec{\alpha} \cup \mathsf{FV}(\vec{\sigma})$. Hence $(\Lambda\alpha r)[\vec{\alpha} := \vec{\sigma}] = \Lambda\alpha.r[\vec{\alpha} := \vec{\sigma}]$. We may assume (2): $\alpha \notin \mathsf{FTV}(\vec{s})$, which gives $(\Lambda\alpha r)' = \Lambda\alpha r'$. We have to show $\Lambda\alpha r' \in \mathsf{SC}_{\forall\alpha\rho}[\vec{\alpha} := \vec{\mathcal{P}}]$. Use (∀-I) for saturated sets: Let σ be a type and $\mathcal{P} \in \mathsf{SAT}_\sigma$. Show that $r'[\alpha := \sigma] \in \mathsf{SC}_\rho[\vec{\alpha},\alpha := \vec{\mathcal{P}},\mathcal{P}]$. We first show that

$$r'[\alpha := \sigma] = r[\vec{\alpha},\alpha := \vec{\sigma},\sigma][\vec{x}^{\vec{\rho}[\vec{\alpha},\alpha:=\vec{\sigma},\sigma]} := \vec{s}]. \tag{$*$}$$

We want to apply Corollary 2.21 (to the interchange law for substitution) in order to get

$$r[\vec{\alpha} := \vec{\sigma}][\vec{x}^{\vec{\rho}[\vec{\alpha}:=\vec{\sigma}]} := \vec{s}][\alpha := \sigma] = r[\vec{\alpha} := \vec{\sigma}][\alpha := \sigma][\vec{x}^{\vec{\rho}[\vec{\alpha}:=\vec{\sigma}]} := \vec{s}[\alpha := \sigma]].$$

If $x^\pi \in \mathsf{FV}(r[\vec{\alpha} := \vec{\sigma}])$, then $\alpha \notin \mathsf{FV}(\pi)$ because $\Lambda\alpha r$ is well-formed and consequently by help of (1) also $\Lambda\alpha.r[\vec{\alpha} := \vec{\sigma}]$. Moreover, $\alpha \notin \mathsf{FV}(\vec{\rho}[\vec{\alpha} := \vec{\sigma}])$ follows by help of (2) and Lemma 2.8 which gives $\mathsf{FV}(\rho_i[\vec{\alpha} := \vec{\sigma}]) \subseteq \mathsf{FTV}(s_i)$ for all i. Hence, Corollary 2.21 applies. By (1) and Lemma 2.19

$$r[\vec{\alpha} := \vec{\sigma}][\alpha := \sigma] = r[\vec{\alpha},\alpha := \vec{\sigma},\sigma].$$

Because $\alpha \notin \mathsf{FV}(\rho_i[\vec{\alpha} := \vec{\sigma}])$ and hence by (1) and Corollary 2.4 which give $\alpha \notin \mathsf{FV}(\rho_i)$, we get

$$\rho_i[\vec{\alpha} := \vec{\sigma}] = \rho_i[\vec{\alpha},\alpha := \vec{\sigma},\sigma].$$

Because $\alpha \notin \mathsf{FTV}(s_i)$, $s_i[\alpha := \sigma] = s_i$. We conclude that $(*)$ holds. Show that

$$r[\vec{\alpha},\alpha := \vec{\sigma},\sigma][\vec{x}^{\vec{\rho}[\vec{\alpha},\alpha:=\vec{\sigma},\sigma]} := \vec{s}] \in \mathsf{SC}_\rho[\vec{\alpha},\alpha := \vec{\mathcal{P}},\mathcal{P}]$$

by induction hypothesis: For all i, $s_i \in \mathsf{SC}_{\rho_i}[\vec{\alpha},\alpha := \vec{\mathcal{P}},\mathcal{P}]$ by Lemma 9.13 because $\alpha \notin \mathsf{FV}(\rho_i)$. Finally we have to check injectivity: Let $x^\pi \in \mathsf{FV}(r) \cup \vec{x}^{\vec{\rho}}$. Then $\alpha \notin \mathsf{FV}(\pi)$: If $x^\pi \in \mathsf{FV}(r)$, this follows from the well-formedness of $\Lambda\alpha r$. If $x^\pi = x_i^{\rho_i}$ for some i, then it follows from $\alpha \notin \mathsf{FV}(\rho_i)$. Therefore $\pi[\vec{\alpha},\alpha := \vec{\sigma},\sigma] = \pi[\vec{\alpha} := \vec{\sigma}]$. Now injectivity ("for $\vec{\alpha},\alpha$") follows immediately from the assumed injectivity ("for $\vec{\alpha}$").
(∀-E) Case $r^{\forall\alpha\rho}\sigma$. $(r\sigma)' = r'(\sigma[\vec{\alpha} := \vec{\sigma}])$. Set $\tau := \sigma[\vec{\alpha} := \vec{\sigma}]$ and $\mathcal{P} := \mathsf{SC}_\sigma[\vec{\alpha} := \vec{\mathcal{P}}]$. Then $\mathcal{P} \in \mathsf{SAT}_\tau$ and we have to show that $r'\tau \in \mathsf{SC}_{\rho[\alpha:=\sigma]}[\vec{\alpha} := \vec{\mathcal{P}}]$. Because $\mathsf{FV}(r) = \mathsf{FV}(r\sigma)$, the induction hypothesis gives $r' \in \mathsf{SC}_{\forall\alpha\rho}[\vec{\alpha} := \vec{\mathcal{P}}]$. By (∀-E) for saturated sets it follows $r'\tau \in \mathsf{SC}_\rho[\vec{\alpha},\alpha := \vec{\mathcal{P}},\mathcal{P}] = \mathsf{SC}_{\rho[\alpha:=\sigma]}[\vec{\alpha} := \vec{\mathcal{P}}]$ by Lemma 9.14[11]. □

Theorem $\mathsf{SN} = \Lambda$.

Proof Take empty lists in the preceding lemma. Because empty substitutions do not affect the types and terms (Corollary 2.2, Lemma 2.11 and Lemma 2.16), the injectivity condition is trivially fulfilled and $r \in \mathsf{SC}_\rho[:=]$[12] $\in \mathsf{SAT}$ which implies $\mathsf{SC}_\rho[:=] \subseteq \mathsf{SN}$ and hence $r \in \mathsf{SN}$[13]. □

Corollary 9.17 Every term of system F is strongly normalizing.

Proof Lemma 9.6. □

[11] We already assumed $\alpha \notin \vec{\alpha}$ in the definition of $\mathsf{SC}_{\forall\alpha\rho}[\vec{\alpha} := \vec{\mathcal{P}}]$.
[12] Perhaps one should prefer to write SC_ρ instead of $\mathsf{SC}_\rho[:=]$.
[13] It might be interesting to check that we never used the induction on the inductively defined set SN. We only used the defining clauses expressing the head reduction strategy.

Corollary 9.18 The function Ω defined in section 2.1.4 is a function from Λ to NF and hence a normalizing function.

Proof Lemma 2.38. □

Corollary 9.19 There is no closed term of type $\forall\alpha\alpha$.

Proof Lemma 2.37. □

9.2 Strong normalization of system eF+ex

In the section following this proof we will see how the well-known impredicative encoding of $+$ and $\exists$ which was already used in the embedding of eF into F and in that of eF+ex into eF may be read off the proof.

9.2.1 Terms in SN are strongly normalizing

Lemma 9.20 If $r, s \in \mathsf{sn}$, then $\langle r, s\rangle \in \mathsf{sn}$.

Proof For example by main induction on $r \in \mathsf{sn}$ and side induction on $s \in \mathsf{sn}$. As for system F the immediate reducts have to be analyzed. □

Lemma 9.21 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 $\langle r, s\rangle\mathsf{R}\vec{s} \in \mathsf{sn}$.

Proof By induction on the ingredients being in sn. □

Lemma 9.22 If $r \in \mathsf{sn}$, then $\mathsf{INL}_\rho r \in \mathsf{sn}$ and $\mathsf{INR}_\rho r \in \mathsf{sn}$.

Proof Similarly. □

Lemma 9.23 If $(\mathsf{INL}_\rho r)st\vec{s}$ is a term and $sr\vec{s} \in \mathsf{sn}$ and $t \in \mathsf{sn}$, then $(\mathsf{INL}_\rho r)st\vec{s} \in \mathsf{sn}$. If $(\mathsf{INR}_\rho r)st\vec{s}$ is a term and $tr\vec{s} \in \mathsf{sn}$ and $s \in \mathsf{sn}$, then $(\mathsf{INR}_\rho r)st\vec{s} \in \mathsf{sn}$.

Proof Similarly. □

Lemma 9.24 If $C_{\exists\alpha\rho,\tau}t$ is a term and $t \in \mathsf{sn}$, then $C_{\exists\alpha\rho,\tau}t \in \mathsf{sn}$.

Proof Similarly. □

Lemma 9.25 If $(C_{\exists\alpha\rho,\tau}t)\mathrm{E}_\exists s\vec{s}$ is a term and $s\tau t\vec{s} \in \mathsf{sn}$, then $(C_{\exists\alpha\rho,\tau}t)\mathrm{E}_\exists s\vec{s} \in \mathsf{sn}$.

Proof Similarly. □

Corollary 9.26 $\mathsf{SN} \subseteq \mathsf{sn}$.

Proof As for Lemma 9.6. Note that also $\mathsf{IN1} \in \mathsf{sn}$. □

A remark is in order: $\mathsf{SN} \subseteq \mathsf{sn}$ is proved by showing that all the defining clauses of SN are closure properties of sn. But we did not show this for (V), ($\to$-I), ($\beta_\to$), ($\forall$-I) and ($\beta_\forall$). Those clauses were only considered for system F. But the systems are organized in such a way as to allow modular proofs. Hence, the proofs in section 9.1.1 hold verbatim for eF+ex. This is a general pattern: In the proof of any lemma needed for the proof of strong normalization we only have to consider the newly introduced type or term constructs because the proof clauses for the original constructs are also valid in the extended system. This is what I called non-quadratic reasoning in section 2.2.3. The meta-convention on extensions stated in that section shall now also cover the above pattern: If in the proof of a statement in the extension of a system for which the same statement has already been proved only clauses are given which correspond to the additional constructs of the extension then the proofs for the original system hold verbatim for the extension.

9.2.2 Saturated sets

Extend the definition for system F of $\mathcal{M}$ being a τ-saturated set by the following clauses:

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

(β_+) If $(\mathsf{INL}_\rho r)st\vec{s}$ is a term and $sr\vec{s} \in \mathcal{M}$ and $t \in \mathsf{SN}$, then $(\mathsf{INL}_\rho r)st\vec{s} \in \mathcal{M}$. If $(\mathsf{INR}_\rho r)st\vec{s}$ is a term and $tr\vec{s} \in \mathcal{M}$ and $s \in \mathsf{SN}$, then $(\mathsf{INR}_\rho r)st\vec{s} \in \mathcal{M}$.

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

Extend the definition of the τ-saturated closure $\mathsf{cl}_\tau(M)$ accordingly by the clauses

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

(β_+) If $(\mathsf{INL}_\rho r)st\vec{s}$ is a term and $sr\vec{s} \in \mathsf{cl}_\tau(M)$ and $t \in \mathsf{SN}$, then $(\mathsf{INL}_\rho r)st\vec{s} \in \mathsf{cl}_\tau(M)$. If $(\mathsf{INR}_\rho r)st\vec{s}$ is a term and $tr\vec{s} \in \mathsf{cl}_\tau(M)$ and $s \in \mathsf{SN}$, then $(\mathsf{INR}_\rho r)st\vec{s} \in \mathsf{cl}_\tau(M)$.

($\beta_\exists$) If $(C_{\exists\alpha\rho,\tau}t)\mathrm{E}_\exists s\vec{s}$ is a term and $s\tau t\vec{s} \in \mathsf{cl}_\tau(M)$, then $(C_{\exists\alpha\rho,\tau}t)\mathrm{E}_\exists s\vec{s} \in \mathsf{cl}_\tau(M)$.

As for system F this is a strictly positive inductive definition. One also shows $\mathsf{cl}_\tau(M) \subseteq \mathsf{SN}_\tau$. Hence, again $\mathsf{cl}_\tau(M)$ is the smallest τ-saturated set containing $M \cap \mathsf{SN}_\tau$.

Calculating with saturated sets

The zero type Define $0_{\mathsf{SAT}} := \mathsf{cl}_0(\emptyset)$.

Lemma 9.27 If $r \in 0_{\mathsf{SAT}}$, ρ type and $\mathcal{M} \in \mathsf{SAT}_\rho$, then $r\mathrm{E}_0\rho \in \mathcal{M}$.

Proof Induction on $r \in \mathsf{cl}_0(\emptyset)$. □

Therefore, we have the following rule:

(0-E) If $r \in 0_{\mathsf{SAT}}$, then $\forall\rho\forall\mathcal{M} \in \mathsf{SAT}_\rho\, r\mathrm{E}_0\rho \in \mathcal{M}$.

The one type Define $1_{\mathsf{SAT}} := \mathsf{cl}_1(\Lambda_1) = \mathsf{SN}_1$. Obviously, we have the following rule:

(1-I) $\mathsf{IN1} \in 1_{\mathsf{SAT}}$.

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

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

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 $\mathcal{N}$.

Lemma 9.28 $M_E \cap \mathsf{SN} \in \mathsf{SAT}$ (which implies $\mathcal{M} \times_E \mathcal{N} = M_E \cap \mathsf{SN}$).

Proof Very easy. One only has to append the symbols L and R to the vector $\vec{s}$. □

Lemma 9.29 $M_I \subseteq \mathsf{SN}$ (which implies $\mathcal{M} \times_I \mathcal{N} \supseteq M_I$). □

Lemma 9.30 $\mathcal{M} \times_I \mathcal{N} \subseteq \mathcal{M} \times_E \mathcal{N}$.

Proof It suffices to show $M_I \subseteq M_E$. This is an application of the rule $(\beta_\times)$ of the definition of saturated sets (with empty $\vec{s}$). □

The preceding three statements immmediately imply the following rules for calculating with $\mathcal{M} \times_I \mathcal{N}$ and $\mathcal{M} \times_E \mathcal{N}$:

($\times_E$-E) If $r \in \mathcal{M} \times_E \mathcal{N}$, then $r\mathsf{L} \in \mathcal{M}$ and $r\mathsf{R} \in \mathcal{N}$.

($\times_I$-E) If $r \in \mathcal{M} \times_I \mathcal{N}$, then $r\mathsf{L} \in \mathcal{M}$ and $r\mathsf{R} \in \mathcal{N}$.

($\times_I$-I) If $r \in \mathcal{M}$ and $s \in \mathcal{N}$, then $\langle r, s\rangle \in \mathcal{M} \times_I \mathcal{N}$.

($\times_E$-I) If $r \in \mathcal{M}$ and $s \in \mathcal{N}$, then $\langle r, s\rangle \in \mathcal{M} \times_E \mathcal{N}$.

In the sequel $\mathcal{M} \times \mathcal{N}$ will denote either $\mathcal{M} \times_I \mathcal{N}$ or $\mathcal{M} \times_E \mathcal{N}$ as long as it does not matter which one is meant, i.e. the use of this set is restricted to the following rules:

(SAT) $\mathcal{M} \times \mathcal{N} \in \mathsf{SAT}_{\rho\times\sigma}$.

($\times$-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 $r\mathsf{R} \in \mathcal{N}$.

It is also possible to produce an example showing $\mathcal{M} \times_I \mathcal{N} \neq \mathcal{M} \times_E \mathcal{N}$: Let $\rho, \sigma := \alpha \times \alpha$, $\mathcal{M} := \mathsf{cl}_{\alpha\times\alpha}(\{\langle t, t\rangle \mathsf{L} | t \in \mathsf{SN}_{\alpha\times\alpha}\})$ and $\mathcal{N} := \mathsf{cl}_{\alpha\times\alpha}(\{\langle t, t\rangle \mathsf{R} | t \in \mathsf{SN}_{\alpha\times\alpha}\})$. It is easy to show that $\langle\langle x^\alpha, x^\alpha\rangle, \langle x^\alpha, x^\alpha\rangle\rangle \in \mathcal{M} \times_E \mathcal{N} \setminus \mathcal{M} \times_I \mathcal{N}$.

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

$$\begin{array}{lcl} M_I & := & \{r \in \Lambda_{\rho+\sigma} | (\exists s \in \mathcal{M}.r = \mathsf{INL}_\sigma s) \vee (\exists s \in \mathcal{N}.r = \mathsf{INR}_\rho s)\} \\ M_E & := & \{r \in \Lambda_{\rho+\sigma} | \forall\tau\forall\mathcal{P} \in \mathsf{SAT}_\tau \forall s \in \mathcal{M} \to \mathcal{P} \forall t \in \mathcal{N} \to \mathcal{P}.r\mathrm{E}_+\tau st \in \mathcal{P}\} \end{array}$$

and set $\mathcal{M} +_I \mathcal{N} := \mathsf{cl}_{\rho+\sigma}(M_I)$ and $\mathcal{M} +_E \mathcal{N} := \mathsf{cl}_{\rho+\sigma}(M_E)$. Hence, $\mathcal{M} +_I \mathcal{N}$ and $\mathcal{M} +_E \mathcal{N}$ are $(\rho + \sigma)$-saturated sets and the construction is isotone in the arguments $\mathcal{M}$ and $\mathcal{N}$. ($\mathcal{M} +_E \mathcal{N}$ is non-strictly positive in $\mathcal{M}$ and $\mathcal{N}$.) Note that we already use the indeterminate construction $\to$ for saturated sets in the definition of M_E.

Lemma 9.31 $M_E \cap \mathsf{SN} \in \mathsf{SAT}$ (which implies $\mathcal{M} +_E \mathcal{N} = M_E \cap \mathsf{SN}$).

Proof Very easy. One only has to append s and t to the vector $\vec{s}$. □

Lemma 9.32 $M_I \subseteq \mathsf{SN}$ (which implies $\mathcal{M} +_I \mathcal{N} \supseteq M_I$). □

Lemma 9.33 $\mathcal{M} +_I \mathcal{N} \subseteq \mathcal{M} +_E \mathcal{N}$.

Proof It suffices to show $M_I \subseteq M_E$. This is an application of the rule (β_+) of the definition of saturated sets (with empty $\vec{s}$) and of ($\to$-E) for saturated sets. □

The preceding three statements immediately imply the following rules for calculating with $\mathcal{M} +_I \mathcal{N}$ and $\mathcal{M} +_E \mathcal{N}$:

($+_E$-E) If $r \in \mathcal{M} +_E \mathcal{N}$, then $\forall\tau\forall\mathcal{P} \in \mathsf{SAT}_\tau \forall s \in \mathcal{M} \to \mathcal{P} \forall t \in \mathcal{N} \to \mathcal{P}.r\mathrm{E}_+\tau st \in \mathcal{P}$.

($+_I$-E) If $r \in \mathcal{M} +_I \mathcal{N}$, then $\forall\tau\forall\mathcal{P} \in \mathsf{SAT}_\tau \forall s \in \mathcal{M} \to \mathcal{P} \forall t \in \mathcal{N} \to \mathcal{P}.r\mathrm{E}_+\tau st \in \mathcal{P}$.

($+_I$-I) If $s \in \mathcal{M}$, then $\mathsf{INL}_\sigma s \in \mathcal{M} +_I \mathcal{N}$. If $s \in \mathcal{N}$, then $\mathsf{INR}_\rho s \in \mathcal{M} +_I \mathcal{N}$.

($+_E$-I) If $s \in \mathcal{M}$, then $\mathsf{INL}_\sigma s \in \mathcal{M} +_E \mathcal{N}$. If $s \in \mathcal{N}$, then $\mathsf{INR}_\rho s \in \mathcal{M} +_E \mathcal{N}$.

In the sequel $\mathcal{M} + \mathcal{N}$ will denote either $\mathcal{M} +_I \mathcal{N}$ or $\mathcal{M} +_E \mathcal{N}$ as long as it does not matter which one is meant, i.e. the use of this set is restricted to the following rules:

(SAT) $\mathcal{M} + \mathcal{N} \in \mathsf{SAT}_{\rho+\sigma}$.

(+-I) If $s \in \mathcal{M}$, then $\mathsf{INL}_\sigma s \in \mathcal{M} + \mathcal{N}$. If $s \in \mathcal{N}$, then $\mathsf{INR}_\rho s \in \mathcal{M} + \mathcal{N}$.

(+-E) If $r \in \mathcal{M} + \mathcal{N}$, then $\forall\tau\forall\mathcal{P} \in \mathsf{SAT}_\tau \forall s \in \mathcal{M} \to \mathcal{P} \forall t \in \mathcal{N} \to \mathcal{P}.r\mathrm{E}_+\tau st \in \mathcal{P}$.

Existential types 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

$$\begin{array}{lcl} M_I & := & \{r \in \Lambda_{\exists\alpha\rho} | \exists\tau\exists\mathcal{P} \in \mathsf{SAT}_\tau \exists t \in \Phi_\tau(\mathcal{P}).r = C_{\exists\alpha\rho,\tau} t\} \\ M_E & := & \{r \in \Lambda_{\exists\alpha\rho} | \forall\sigma\forall\mathcal{N} \in \mathsf{SAT}_\sigma \forall s \in (\forall\mathcal{P}.\Phi(\mathcal{P}) \to \mathcal{N}).r\mathrm{E}_\exists\sigma s \in \mathcal{N}\} \end{array}$$

and set $\exists_I(\Phi) := \mathsf{cl}_{\exists\alpha\rho}(M_I)$ and $\exists_E(\Phi) := \mathsf{cl}_{\exists\alpha\rho}(M_E)$. Hence, $\exists_I(\Phi)$ and $\exists_E(\Phi)$ are $\exists\alpha\rho$-saturated sets and the construction is (pointwise) isotone in the argument Φ. Note that we use the indeterminate $\to$ and $\forall$ for saturated sets in the definition of M_E.

Lemma 9.34 $M_E \cap \mathsf{SN} \in \mathsf{SAT}$ (which implies $\exists_E(\Phi) = M_E \cap \mathsf{SN}$).

Proof Append s to $\vec{s}$. □

Lemma 9.35 $M_I \subseteq \mathsf{SN}$ (which implies $\exists_I(\Phi) \supseteq M_I$). □

Lemma 9.36 $\exists_I(\Phi) \subseteq \exists_E(\Phi)$.

Proof It suffices to show $M_I \subseteq M_E$. This is an application of the rule $(\beta_\exists)$ of the definition of saturated sets (with empty $\vec{s}$) and of ($\forall$-E) and ($\to$-E) for saturated sets. □

We get the following rules:

($\exists_E$-E) If $r \in \exists_E(\Phi)$, then $\forall\sigma\forall\mathcal{N} \in \mathsf{SAT}_\sigma \forall s \in (\forall\mathcal{P}.\Phi(\mathcal{P}) \to \mathcal{N}).r\mathrm{E}_\exists\sigma s \in \mathcal{N}$.

($\exists_I$-E) If $r \in \exists_I(\Phi)$, then $\forall\sigma\forall\mathcal{N} \in \mathsf{SAT}_\sigma \forall s \in (\forall\mathcal{P}.\Phi(\mathcal{P}) \to \mathcal{N}).r\mathrm{E}_\exists\sigma s \in \mathcal{N}$.

($\exists_I$-I) If $t \in \Phi_\tau(\mathcal{P})$ for some $\mathcal{P} \in \mathsf{SAT}_\tau$, then $C_{\exists\alpha\rho,\tau} t \in \exists_I(\Phi)$.

($\exists_E$-I) If $t \in \Phi_\tau(\mathcal{P})$ for some $\mathcal{P} \in \mathsf{SAT}_\tau$, then $C_{\exists\alpha\rho,\tau} t \in \exists_E(\Phi)$.

From now on $\exists(\Phi)$ shall denote $\exists_I(\Phi)$ or $\exists_E(\Phi)$ if only the following rules are used.

(SAT) $\exists(\Phi) \in \mathsf{SAT}_{\exists\alpha\rho}$.

($\exists$-I) If $t \in \Phi_\tau(\mathcal{P})$ for some $\mathcal{P} \in \mathsf{SAT}_\tau$, then $C_{\exists\alpha\rho,\tau} t \in \exists(\Phi)$.

($\exists$-E) If $r \in \exists(\Phi)$, then $\forall\sigma\forall\mathcal{N} \in \mathsf{SAT}_\sigma \forall s \in (\forall\mathcal{P}.\Phi(\mathcal{P}) \to \mathcal{N}).r\mathrm{E}_\exists\sigma s \in \mathcal{N}$.

Computability predicates

Extend the definition of $\mathsf{SC}_\rho[\vec{\alpha} := \vec{\mathcal{P}}]$ by the following clauses:

(0) $\mathsf{SC}_0[\vec{\alpha} := \vec{\mathcal{P}}] := 0_{\mathsf{SAT}}$.

(1) $\mathsf{SC}_1[\vec{\alpha} := \vec{\mathcal{P}}] := 1_{\mathsf{SAT}}$.

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

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

($\exists$) $\mathsf{SC}_{\exists\alpha\rho}[\vec{\alpha} := \vec{\mathcal{P}}] := \exists(\Phi)$ with $\Phi = (\Phi_\sigma)_\sigma$, where $\Phi_\sigma : \mathsf{SAT}_\sigma \to \mathsf{SAT}_{\rho[\vec{\alpha},\alpha:=\vec{\sigma},\sigma]}$ is defined by $\Phi_\sigma(\mathcal{P}) := \mathsf{SC}_\rho[\vec{\alpha},\alpha := \vec{\mathcal{P}},\mathcal{P}]$ (with the same assumptions as in the definition of strong computability for universal types in order to guarantee type-correctness).

The definition in clause ($\exists$) will be written more intuitively as follows:

$$\mathsf{SC}_{\exists\alpha\rho}[\vec{\alpha} := \vec{\mathcal{P}}] := \exists\mathcal{P}.\mathsf{SC}_\rho[\vec{\alpha},\alpha := \vec{\mathcal{P}},\mathcal{P}].$$

Evidently, the choice whether one uses the introduction-based or the elimination-based construction for saturated sets has to be consistent in the sense that if one uses e. g. $\exists_E(\Phi)$ with M_E defined by reference to $\to_I$, then $\mathsf{SC}_{\rho\to\sigma}[\vec{\alpha} := \vec{\mathcal{P}}]$ also has to be defined by means of $\to_I$. As for system F we therefore assume that this choice is made once for every type construct and then used throughout.
With this fixed choice, it is obvious that coincidence and substitution lemma (Lemma 9.13 and Lemma 9.14) extend to $\mathsf{eF{+}ex}$.
Now we extend Lemma 9.16 to $\mathsf{eF{+}ex}$:

Lemma 9.37 Let r^ρ be a term, $\vec{x}^{\vec{\rho}}$ a list of variables and $\vec{s}$ an equally long list of terms such that $\vec{s} \in \mathsf{SC}_{\vec{\rho}}[\vec{\alpha} := \vec{\mathcal{P}}^{\vec{\sigma}}]$ for some $\vec{\mathcal{P}} \subseteq \mathsf{SAT}$. Assume that

$$\forall x\forall\pi\forall\pi'.(x^\pi, x^{\pi'} \in \mathsf{FV}(r) \cup \vec{x}^{\vec{\rho}}) \wedge (\pi[\vec{\alpha} := \vec{\sigma}] = \pi'[\vec{\alpha} := \vec{\sigma}]) \Rightarrow \pi = \pi'.$$

Then $r[\vec{\alpha} := \vec{\sigma}][\vec{x}^{\vec{\rho}[\vec{\alpha}:=\vec{\sigma}]} := \vec{s}] \in \mathsf{SC}_\rho[\vec{\alpha} := \vec{\mathcal{P}}]$.

Proof Induction on r. Again abbreviate $r[\vec{\alpha} := \vec{\sigma}][\vec{x}^{\vec{\rho}[\vec{\alpha}:=\vec{\sigma}]} := \vec{s}]$ by r' (for any term r). The assumption in the statement will again be referenced to as "injectivity". We only have to consider the term generation rules corresponding to the types and type constructions newly introduced in $\mathsf{eF{+}ex}$.
(0-E) Case $r^0\rho$. $(r\rho)' = r'(\rho[\vec{\alpha} := \vec{\sigma}])$. Because $\mathsf{FV}(r\rho) = \mathsf{FV}(r)$, injectivity holds also for r. By induction hypothesis, $r' \in \mathsf{SC}_0[\vec{\alpha} := \vec{\mathcal{P}}] = 0_{\mathsf{SAT}}$. By (0-E) for saturated sets (the reference to the named rules will in the sequel always mean those for saturated sets), $r'(\rho[\vec{\alpha} := \vec{\sigma}]) \in \mathsf{SC}_\rho[\vec{\alpha} := \vec{\mathcal{P}}]$.

(1-I) Case $\mathsf{IN1}$. $\mathsf{IN1}' = \mathsf{IN1}$. By (1-I), $\mathsf{IN1} \in 1_{\mathsf{SAT}} = \mathsf{SC}_1[\vec{\alpha} := \vec{\mathcal{P}}]$.
($\times$-I) Case $\langle r^\rho, s^\sigma\rangle$. $\langle r, s\rangle' = \langle r', s'\rangle$. Because $\mathsf{FV}(\langle r, s\rangle) = \mathsf{FV}(r) \cup \mathsf{FV}(s)$, injectivity also holds for r and s. By induction hypothesis, $r' \in \mathsf{SC}_\rho[\vec{\alpha} := \vec{\mathcal{P}}]$ and $s' \in \mathsf{SC}_\sigma[\vec{\alpha} := \vec{\mathcal{P}}]$. By ($\times$-I) we conclude $\langle r', s'\rangle \in \mathsf{SC}_{\rho\times\sigma}[\vec{\alpha} := \vec{\mathcal{P}}]$.
($\times$-E) Case $r^{\rho\times\sigma}\mathsf{L}$. $(r\mathsf{L})' = r'\mathsf{L}$. Because $\mathsf{FV}(r\mathsf{L}) = \mathsf{FV}(r)$, injectivity also holds for r. By induction hypothesis, $r' \in \mathsf{SC}_{\rho\times\sigma}[\vec{\alpha} := \vec{\mathcal{P}}]$. By ($\times$-E), we get $r'\mathsf{L} \in \mathsf{SC}_\rho[\vec{\alpha} := \vec{\mathcal{P}}]$. The case $r^{\rho\times\sigma}\mathsf{R}$ is similar.
($+$-I) Case $\mathsf{INL}_\sigma r^\rho$. $(\mathsf{INL}_\sigma r)' = \mathsf{INL}_{\sigma[\vec{\alpha}:=\vec{\sigma}]} r'$. Because $\mathsf{FV}(\mathsf{INL}_\sigma r) = \mathsf{FV}(r)$, injectivity also holds for r. By induction hypothesis, $r' \in \mathsf{SC}_\rho[\vec{\alpha} := \vec{\mathcal{P}}]$. By ($+$-I), $\mathsf{INL}_{\sigma[\vec{\alpha}:=\vec{\sigma}]} r' \in \mathsf{SC}_{\rho+\sigma}[\vec{\alpha} := \vec{\mathcal{P}}]$. The case $\mathsf{INR}_\rho r^\sigma$ is similar.
($+$-E) Case $r^{\rho+\sigma}\mathrm{E}_+\tau s^{\rho\to\tau} t^{\sigma\to\tau}$. $(rst)' = r'\mathrm{E}_+\tau[\vec{\alpha} := \vec{\sigma}]s't'$. Because $\mathsf{FV}(rst) = \mathsf{FV}(r) \cup \mathsf{FV}(s) \cup \mathsf{FV}(t)$, injectivity holds also for r, s and t. By induction hypothesis we have $r' \in \mathsf{SC}_{\rho+\sigma}[\vec{\alpha} := \vec{\mathcal{P}}]$, $s' \in \mathsf{SC}_{\rho\to\tau}[\vec{\alpha} := \vec{\mathcal{P}}]$ and $t' \in \mathsf{SC}_{\sigma\to\tau}[\vec{\alpha} := \vec{\mathcal{P}}]$. Hence, by ($+$-E): $r'\mathrm{E}_+\tau[\vec{\alpha} := \vec{\sigma}]s't' \in \mathsf{SC}_\tau[\vec{\alpha} := \vec{\mathcal{P}}]$.
($\exists$-I) Case $C_{\exists\alpha\rho,\tau} t^{\rho[\alpha:=\tau]}$. We may assume that $\alpha \notin \vec{\alpha} \cup \mathsf{FV}(\vec{\sigma})$. Therefore, we have that $(C_{\exists\alpha\rho,\tau} t)' = C_{\exists\alpha.\rho[\vec{\alpha}:=\vec{\sigma}],\tau[\vec{\alpha}:=\vec{\sigma}]} t'$. Show $C_{\exists\alpha.\rho[\vec{\alpha}:=\vec{\sigma}],\tau[\vec{\alpha}:=\vec{\sigma}]} t' \in \mathsf{SC}_{\exists\alpha\rho}[\vec{\alpha} := \vec{\mathcal{P}}]$ by use of ($\exists$-I). It suffices to show

$$t' \in \mathsf{SC}_\rho[\vec{\alpha},\alpha := \vec{\mathcal{P}}, \mathsf{SC}_\tau[\vec{\alpha} := \vec{\mathcal{P}}]].$$

By the substitution lemma we have

$$\mathsf{SC}_\rho[\vec{\alpha},\alpha := \vec{\mathcal{P}}, \mathsf{SC}_\tau[\vec{\alpha} := \vec{\mathcal{P}}]] = \mathsf{SC}_{\rho[\alpha:=\tau]}[\vec{\alpha} := \vec{\mathcal{P}}].$$

Hence it suffices to show $t' \in \mathsf{SC}_{\rho[\alpha:=\tau]}[\vec{\alpha} := \vec{\mathcal{P}}]$ which follows from the induction hypothesis because injectivity also holds for t due to $\mathsf{FV}(C_{\exists\alpha\rho,\tau} t) = \mathsf{FV}(t)$.
($\exists$-E) Case $r^{\exists\alpha\rho}\mathrm{E}_\exists\sigma s^{\forall\alpha.\rho\to\sigma}$ (with $\alpha \notin \mathsf{FV}(\sigma)$). $(rs)' = r'\mathrm{E}_\exists\sigma[\vec{\alpha} := \vec{\sigma}]s'$. Because $\mathsf{FV}(rs) = \mathsf{FV}(r) \cup \mathsf{FV}(s)$, injectivity also holds for r and s. By induction hypothesis $r' \in \mathsf{SC}_{\exists\alpha\rho}[\vec{\alpha} := \vec{\mathcal{P}}]$ and $s' \in \mathsf{SC}_{\forall\alpha.\rho\to\sigma}[\vec{\alpha} := \vec{\mathcal{P}}]$. Show that $r'\mathrm{E}_\exists\sigma[\vec{\alpha} := \vec{\sigma}]s' \in \mathsf{SC}_\sigma[\vec{\alpha} := \vec{\mathcal{P}}]$ by help of ($\exists$-E). It suffices to show that $s' \in \forall\mathcal{P}.\mathsf{SC}_\rho[\vec{\alpha},\alpha := \vec{\mathcal{P}},\mathcal{P}] \to \mathsf{SC}_\sigma[\vec{\alpha} := \vec{\mathcal{P}}]$. This holds by definition of strong computability and by the coincidence lemma (recall that $\alpha \notin \mathsf{FV}(\sigma)$). □

As for system F we get as corollaries that $\mathsf{SN} = \Lambda$, that every term is strongly normalizing, that Ω is a normalizing function and that there is no closed term of type $\forall\alpha\alpha$. Lemma 2.46 also gives us that there is no closed term of type 0.
Note that every definition and lemma in this section can be done for system eF by simply leaving out the clauses referring to the existential quantifier. The definitions in the next section will be based on these concepts for eF.

9.3 Extracting embeddings from normalization proofs

The proofs of strong normalization of F and $\mathsf{eF{+}ex}$ are very close to a formalization. There is no analysis of reducts and only the introduction rule and the elimination rule for the constructions of saturated sets corresponding to the type formation rules are used in the final soundness lemma 9.16 (strong computability under substitution). It appears that there is a considerable overhead for dealing with substitution issues. After all, substitution is the governing principle of system F. Therefore, program extraction from intuitionistic proofs seems quite promising because it divides the bookkeeping part from the interesting combinatorial problem solved. Modified realizability will again be used informally.
Let us first study the case of sum types.

$$\mathcal{M} +_I \mathcal{N} = \mathsf{cl}_{\rho+\sigma}(\{r \in \Lambda_{\rho+\sigma} | (\exists s \in \mathcal{M}.r = \mathsf{INL}_\sigma s) \vee (\exists s \in \mathcal{N}.r = \mathsf{INR}_\rho s)\})$$

If $\mathcal{M}$ already had the extracted type ρ' and $\mathcal{N}$ had extracted type σ', we would get as extracted type of the predicate $\mathcal{M} +_I \mathcal{N}$ the type $\rho' + \sigma'$: The closure cl is responsible for closing comprehension predicates and $r = \mathsf{INL}_\sigma s$ and $r = \mathsf{INR}_\rho s$ carry no essential information. What counts is only the extracted types of the predicates involved.

$$\mathcal{M} +_E \mathcal{N} = \mathsf{cl}_{\rho+\sigma}(\{r \in \Lambda_{\rho+\sigma} | \forall \tau \forall \mathcal{P} \in \mathsf{SAT}_\tau \forall s \in \mathcal{M} \to \mathcal{P} \forall t \in \mathcal{N} \to \mathcal{P}. r\mathrm{E}_+ \tau s t \in \mathcal{P}\})$$

This time we would get as extracted type of the predicate $\mathcal{M} +_E \mathcal{N}$ the type $\forall \alpha.(\rho' \to \alpha) \to (\sigma' \to \alpha) \to \alpha$ with $\alpha \notin \mathsf{FV}(\rho') \cup \mathsf{FV}(\sigma')$: The universal quantifier does not come from $\forall \tau$ but from $\forall \mathcal{P} \in \mathsf{SAT}_\tau$ and the extracted type of $\mathcal{M} \to \mathcal{P}$ is $\rho' \to \alpha$ regardless of the decision whether $\mathcal{M} \to \mathcal{P}$ is $\mathcal{M} \to_I \mathcal{P}$ or $\mathcal{M} \to_E \mathcal{P}$. Why? Because clearly the extracted type is the same in both cases. The same effect happens with $\times$: If $\mathcal{M}$ has extracted type ρ' and $\mathcal{N}$ has extracted type σ', then $\mathcal{M} \times_I \mathcal{N}$ and $\mathcal{M} \times_E \mathcal{N}$ have extracted type $\rho' \times \sigma'$. Also the extracted type of 0_{SAT} is 0 (corresponding to the least element of SAT_0) and that of 1_{SAT} is 1 (corresponding to the greatest element of SAT_1).
We want to reconstruct the part of the embedding of eF into F pertaining to $+$. The proof of the lemma stating that $\mathcal{M} +_I \mathcal{N} \subseteq \mathcal{M} +_E \mathcal{N}$ goes as follows: It suffices to show $M_I \subseteq M_E$ (nothing combinatorial). There are two cases: First let $r \in \mathcal{M}$. We have to show that $\forall \tau \forall \mathcal{P} \in \mathsf{SAT}_\tau \forall s \in \mathcal{M} \to \mathcal{P} \forall t \in \mathcal{N} \to \mathcal{P}. \mathsf{INL}_\sigma r \mathrm{E}_+ \tau s t \in \mathcal{P}$. Let τ be a type, $\mathcal{P} \in \mathsf{SAT}_\tau$, $s \in \mathcal{M} \to \mathcal{P}$ and $t \in \mathcal{N} \to \mathcal{P}$. We have to show $\mathsf{INL}_\sigma r \mathrm{E}_+ \tau s t \in \mathcal{P}$. Because $\mathcal{P}$ is saturated (bookkeeping!) it suffices to show $sr \in \mathcal{P}$ and $t \in \mathsf{SN}$ (bookkeeping). Because of ($\to$-E) for saturated sets, applied to $s \in \mathcal{M} \to \mathcal{P}$ and $r \in \mathcal{M}$, finally $sr \in \mathcal{P}$. Clearly, we may read off this proof the term $\Lambda \alpha \lambda x^{\rho' \to \alpha} \lambda y^{\sigma' \to \alpha}. x r'$ if we assume that r' is the extracted term of "$r \in \mathcal{M}$". This was exactly the term $(\mathsf{INL}_\sigma r^\rho)'$ in the embedding of eF into F and it is one half of the proof which gives the rule ($+_E$-I) not counting the proof of ($+_I$-I) which is only done by checking something without combinatorial content. The proof of ($+_E$-E) is no better and hence explains why the clause ($+$-E) of the embedding of eF into F is so simple.
Consider $\exists_I(\Phi)$. If Φ has $\lambda \alpha \rho'$ as extracted type, then $\exists_I(\Phi)$ has extracted type $\exists \alpha \rho'$. This does not give an interesting encoding. But $\exists_E(\Phi)$ has extracted type $\forall \beta.(\forall \alpha.\rho' \to \beta) \to \beta$ for $\beta \notin \{\alpha\} \cup \mathsf{FV}(\rho')$ because the extraction read off the definitions for the universal quantifier is in both cases the homomorphic rule. And this is exactly the crucial clause in the embedding of eF+ex into eF. The proof that $\exists_I(\Phi) \subseteq \exists_E(\Phi)$ contains the idea for the clause ($\exists$-I) of that embedding.
The open question is: Do those encodings read off the proof by saturated sets always preserve β-reduction, i. e., are they in fact embeddings?
In the next sections we will see many more proofs where the method works.

9.4 Strong normalization of MeIT–it and coMeIT–it

For MeIT–it and for coMeIT–it we give an introduction-based and an elimination-based proof. In contrast to the situation of eF following both possibilities of defining a fixed-point construction for saturated sets is more work than only following one of them. Nevertheless, it seems worthwile studying the differences of the proofs of Lemma 9.44 and Lemma 9.46.

9.4.1 Strong normalization of MeIT–it

Lemma 9.38 If $t \in \mathsf{sn}$, then $C_{\mu\alpha\rho} t \in \mathsf{sn}$.

Proof Induction on $t \in \mathsf{sn}$. □

Lemma 9.39 If $s(\mu\alpha\rho)(\lambda y^{\mu\alpha\rho} y)(\lambda x^{\mu\alpha\rho}. x \mathrm{E}^+_\mu \sigma s) t \vec{s} \in \mathsf{sn}$, $x^{\mu\alpha\rho} \notin \mathsf{FV}(s)$ and $(C_{\mu\alpha\rho} t)\mathrm{E}^+_\mu \sigma s \vec{s}$ is a term, then $(C_{\mu\alpha\rho} t)\mathrm{E}^+_\mu \sigma s \vec{s} \in \mathsf{sn}$.

Proof Induction on $s(\mu\alpha\rho)(\lambda y^{\mu\alpha\rho} y)(\lambda x^{\mu\alpha\rho}. x \mathrm{E}^+_\mu \sigma s) t \vec{s} \in \mathsf{sn}^+$ by the usual analysis of the immediate reducts of $(C_{\mu\alpha\rho} t)\mathrm{E}^+_\mu \sigma s \vec{s}$. (Note that we need sn^+ because s occurs twice in the canonical reduct of $(C_{\mu\alpha\rho} t)\mathrm{E}^+_\mu \sigma s \vec{s}$.) □

Corollary 9.40 $\mathsf{SN} \subseteq \mathsf{sn}$.

Proof The same as for Lemma 9.6. □

Saturated sets for MeIT–it

Extend the definition for system eF of $\mathcal{M}$ being a τ-saturated set by the following clause:

(β^+_μ) If $s(\mu\alpha\rho)(\lambda y^{\mu\alpha\rho} y)(\lambda x^{\mu\alpha\rho}. x \mathrm{E}^+_\mu \sigma s) t \vec{s} \in \mathcal{M}$, $x^{\mu\alpha\rho} \notin \mathsf{FV}(s)$ and $(C_{\mu\alpha\rho} t)\mathrm{E}^+_\mu \sigma s \vec{s}$ is a term, then $(C_{\mu\alpha\rho} t)\mathrm{E}^+_\mu \sigma s \vec{s} \in \mathcal{M}$.

Extend the definition of the τ-saturated closure $\mathsf{cl}_\tau(M)$ accordingly by the clause

(β^+_μ) If $s(\mu\alpha\rho)(\lambda y^{\mu\alpha\rho} y)(\lambda x^{\mu\alpha\rho}. x \mathrm{E}^+_\mu \sigma s) t \vec{s} \in \mathsf{cl}_\tau(M)$, $x^{\mu\alpha\rho} \notin \mathsf{FV}(s)$ and $(C_{\mu\alpha\rho} t)\mathrm{E}^+_\mu \sigma s \vec{s}$ is a term, then $(C_{\mu\alpha\rho} t)\mathrm{E}^+_\mu \sigma s \vec{s} \in \mathsf{cl}_\tau(M)$.

Again $\mathsf{cl}_\tau(M)$ is the smallest τ-saturated set containing $M \cap \mathsf{SN}_\tau$.

Calculating with saturated sets for MeIT–it

From now on we always adopt the same operator precedences for constructions of saturated sets as for types, e. g. $\mathcal{P} \to \mathcal{Q} \to \mathcal{M}$ denotes $\mathcal{P} \to (\mathcal{Q} \to \mathcal{M})$.
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 for $\mathcal{M} \in \mathsf{SAT}_{\mu\alpha\rho}$

$$\begin{aligned} M_I(\mathcal{M}) &:= \{r \in \Lambda_{\mu\alpha\rho} | \exists t \in \Phi_{\mu\alpha\rho}(\mathcal{M}). r = C_{\mu\alpha\rho} t\} \\ M_E(\mathcal{M}) &:= \{r \in \Lambda_{\mu\alpha\rho} | \forall \sigma \forall \mathcal{N} \in \mathsf{SAT}_\sigma \forall s \in \forall \mathcal{P}.(\mathcal{P} \to \mathcal{M}) \to (\mathcal{P} \to \mathcal{N}) \to \Phi(\mathcal{P}) \to \mathcal{N}. rs \in \mathcal{N}\} \end{aligned}$$

and $\Psi_I : \mathsf{SAT}_{\mu\alpha\rho} \to \mathsf{SAT}_{\mu\alpha\rho}$ by setting

$$\Psi_I(\mathcal{M}) := \mathsf{cl}_{\mu\alpha\rho}(M_I(\mathcal{M}))$$

and

$$\mu_E(\Phi) := \text{ any fixed point of } \Psi_E : \mathsf{SAT}_{\mu\alpha\rho} \to \mathsf{SAT}_{\mu\alpha\rho},$$

defined by

$$\Psi_E(\mathcal{M}) := \mathsf{cl}_{\mu\alpha\rho}(M_E(\mathcal{M})).$$

Because Ψ_E is monotone (it is even non-strictly positive) and $\mathsf{SAT}_{\mu\alpha\rho}$ is a complete lattice, such a fixed point exists. Hence, $\mu_E(\Phi)$ is $\mu\alpha\rho$-saturated. If we took the least fixed point for $\mu_E(\Phi)$, $\mu_E(\Phi)$ would be (pointwise) isotone in the argument Φ. If $\Phi_{\mu\alpha\rho}$ were monotone, Ψ_I would also be monotone and we could define $\mu_I(\Phi)$ as the least fixed point of Ψ_I. Unfortunately, in the definition of the computability predicates we cannot ensure the monotonicity of $\Phi_{\mu\alpha\rho}$: Let $\rho := \alpha \to 1$. (Hence, $\lambda\alpha\rho$ is trivially monotone although it is not positive.) Assume that $\to$ (for saturated sets) is $\to_E$. In order to define $\mathsf{SC}_{\mu\alpha\rho}$ we have to study $\Phi_{\mu\alpha\rho}$, defined by

$$\Phi_{\mu\alpha\rho}(\mathcal{M}) := \mathsf{SC}_\rho[\vec{\alpha}, \alpha := \vec{\mathcal{P}}, \mathcal{M}] := \mathsf{SC}_\alpha[\vec{\alpha}, \alpha := \vec{\mathcal{P}}, \mathcal{M}] \to_E \mathsf{SC}_1[\vec{\alpha}, \alpha := \vec{\mathcal{P}}, \mathcal{M}] := \mathcal{M} \to_E \mathsf{SN}_1.$$

Let $\mathcal{M} \subseteq \mathcal{M}'$ and $r \in \Phi_{\mu\alpha\rho}(\mathcal{M})$. How could we show $r \in \Phi_{\mu\alpha\rho}(\mathcal{M}')$? $r \in \mathsf{SN}$ is clear. Let $s \in \mathcal{M}'$. We would have to show $r\mathrm{E}_{\to}s \in \mathsf{SN}_1$. By assumption on r we have that $r\mathrm{E}_{\to}t \in \mathsf{SN}_1$ for $t \in \mathcal{M} \subseteq \mathcal{M}'$. Hence, we cannot use this assumption. However, if we already knew that every term is in SN, we could infer that $\Phi_{\mu\alpha\rho}$ is constantly equal to $\mathsf{SN}_{\mu\alpha\rho\to 1}$ and hence monotone. But we are about to proving $\Lambda = \mathsf{SN}$![14]
Nevertheless, there is an introduction-based approach which works. Define for $\mathcal{M} \in \mathsf{SAT}_{\mu\alpha\rho}$

$$M_I^{\supseteq}(\mathcal{M}) \ := \ \{r \in \Lambda_{\mu\alpha\rho} | \exists \mathcal{M}' \in \mathsf{SAT}_{\mu\alpha\rho}.\mathcal{M}' \subseteq \mathcal{M} \wedge \exists t \in \Phi_{\mu\alpha\rho}(\mathcal{M}').r = C_{\mu\alpha\rho}t\}$$

and set

$$\mu_I(\Phi) := \text{ the least fixed point of } \Psi_I^{\supseteq} : \mathsf{SAT}_{\mu\alpha\rho} \to \mathsf{SAT}_{\mu\alpha\rho},$$

defined by

$$\Psi_I^{\supseteq}(\mathcal{M}) := \mathsf{cl}_{\mu\alpha\rho}(M_I^{\supseteq}(\mathcal{M})).$$

Because $\Psi_I^{\supseteq}$ is monotone (is is even strictly positive), such a fixed point exists. It is obvious that $\Psi_I^{\supseteq} \supseteq \Psi_I$ (pointwise). Therefore any pre-fixed-point of $\Psi_I^{\supseteq}$ is also a pre-fixed-point of Ψ_I. Note that for $\Psi_I^{\supseteq}$ only $\Phi_{\mu\alpha\rho}$ is needed. However, for ease of presentation also in this case the existence of the family $(\Phi_\tau)_\tau$ is assumed.
Using informal modified realizability the above definitions motivate two encodings of the types: The definition of $\mu_I(\Phi)$ gives rise[15] to

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

for $\beta \notin \{\alpha\} \cup \mathsf{FV}(\rho)$, hence $(\mu\alpha\rho)' = \mathsf{spos}(\mu\alpha\rho')$ which is the encoding used for embedding UVIT (and hence also MeIT) into MePIT (in section 6.2.3).
The definition of $\mu_E(\Phi)$ suggests[16] the following

$$(\mu\alpha\rho)' := \hat{\mu}\beta\forall\gamma.(\forall\alpha.(\alpha \to \beta) \to (\alpha \to \gamma) \to \rho' \to \gamma) \to \gamma$$

where $\hat{\mu}$ indicates that a fixed-point type is used instead of an inductive type (see appendix B). Following the proofs of this section one may also read off embeddings: In the first case into MePIT, in the second[17] into a system of non-interleaving positive fixed-point types. That both preserve reduction has to be checked due to the lack of a general theorem saying when the extracted encodings are in fact embeddings.

Lemma 9.41 $M_E(\mathcal{M}) \cap \mathsf{SN} \in \mathsf{SAT}$ (which implies $\Psi_E(\mathcal{M}) = M_E(\mathcal{M}) \cap \mathsf{SN}$).

Proof Straightforward. □

Lemma 9.42 $M_I(\mathcal{M}) \subseteq \mathsf{SN}$ (which implies $\Psi_I(\mathcal{M}) \supseteq M_I(\mathcal{M})$). □

[14] We could partly solve the problem by considering only $\mu\alpha\rho$ with $\alpha \in \mathsf{Pos}(\rho)$ and monotone $\Phi_{\mu\alpha\rho}$ and later proving simultaneously with the definition of the computability predicates that the function $\mathcal{P} \mapsto \mathsf{SC}_\rho[\vec{\alpha}, \alpha := \vec{\mathcal{P}}, \mathcal{P}]$ is monotone for $\alpha \in \mathsf{Pos}(\rho)$ and antitone for $\alpha \in \mathsf{Neg}(\rho)$. We would thus get strong normalization only for the original system of Mendler (coinductive types not taken into account) and were not allowed to infer strong normalization of IMIT from this result.

[15] The existence of $\mathcal{M}'$ is reflected by $\exists\beta$, the inclusion by $\to$, the conjunction by $\times$ and $t \in \Phi_{\mu\alpha\rho}(\mathcal{M}')$ by $\rho'[\alpha := \beta]$. $r = C_{\mu\alpha\rho}t$ has no computational content and hence is not represented.

[16] $\forall\gamma$ reflects $\forall\sigma\forall\mathcal{N} \in \mathsf{SAT}_\sigma$.

[17] For the translation of the introduction rule one uses the slightly easier proof that $\mu_E(\Phi)$ is a pre-fixed-point of Ψ_I instead of Lemma 9.44.

Lemma 9.43 Let $\mathcal{M}$ be a $\mu\alpha\rho$-saturated set.
$\mathcal{M}$ is a pre-fixed-point of Ψ_I iff for all $t \in \Phi_{\mu\alpha\rho}(\mathcal{M})$, we have that $C_{\mu\alpha\rho}t \in \mathcal{M}$.
$\mathcal{M}$ is a post-fixed-point of Ψ_E iff for all $r \in \mathcal{M}$, types σ, $\mathcal{N} \in \mathsf{SAT}_\sigma$ and for all terms $s \in \forall\mathcal{P}.(\mathcal{P} \to \mathcal{M}) \to (\mathcal{P} \to \mathcal{N}) \to \Phi(\mathcal{P}) \to \mathcal{N}$, we have $r\mathrm{E}_\mu^+\sigma s \in \mathcal{N}$.

Proof Use the two preceding lemmas. □

Lemma 9.44 $\mu_E(\Phi)$ is a pre-fixed-point of $\Psi_I^{\supseteq}$.

Proof We have to show that

$$\Psi_I^{\supseteq}(\mu_E(\Phi)) \subseteq \mu_E(\Phi).$$

Because $\mu_E(\Phi)$ is a pre-fixed-point of Ψ_E, it suffices to show

$$\Psi_I^{\supseteq}(\mu_E(\Phi)) \subseteq \Psi_E(\mu_E(\Phi)).$$

Because $\mathsf{cl}_{\mu\alpha\rho}$ is monotone, it suffices to show

$$M_I^{\supseteq}(\mu_E(\Phi)) \subseteq M_E(\mu_E(\Phi)).$$

Let $\mathcal{M} \in \mathsf{SAT}_{\mu\alpha\rho}$ with $\mathcal{M} \subseteq \mu_E(\Phi)$. Let $t \in \Phi_{\mu\alpha\rho}(\mathcal{M})$. We have to show

$$C_{\mu\alpha\rho}t \in M_E(\mu_E(\Phi)).$$

Let σ be a type, $\mathcal{N} \in \mathsf{SAT}_\sigma$ and $s \in \forall\mathcal{P}.(\mathcal{P} \to \mu_E(\Phi)) \to (\mathcal{P} \to \mathcal{N}) \to \Phi(\mathcal{P}) \to \mathcal{N}$. We have to show that

$$C_{\mu\alpha\rho}t\mathrm{E}_\mu^+\sigma s \in \mathcal{N}.$$

Because $\mathcal{N}$ is saturated it suffices to show

$$s(\mu\alpha\rho)(\lambda y^{\mu\alpha\rho}y)(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma s)t \in \mathcal{N}.$$

Because of ($\to$-E) for saturated sets (in the sequel the named rules will always be those for saturated sets) and $t \in \Phi_{\mu\alpha\rho}(\mathcal{M})$ it suffices to show

$$s(\mu\alpha\rho)(\lambda y^{\mu\alpha\rho}y)(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma s) \in \Phi_{\mu\alpha\rho}(\mathcal{M}) \to \mathcal{N}.$$

Because of ($\forall$-E) we have

$$s(\mu\alpha\rho) \in (\mathcal{M} \to \mu_E(\Phi)) \to (\mathcal{M} \to \mathcal{N}) \to \Phi_{\mu\alpha\rho}(\mathcal{M}) \to \mathcal{N}.$$

By help of ($\to$-I) we know $\lambda y^{\mu\alpha\rho}y \in \mathcal{M} \to \mu_E(\Phi)$. Hence by ($\to$-E)

$$s(\mu\alpha\rho)(\lambda y^{\mu\alpha\rho}y) \in (\mathcal{M} \to \mathcal{N}) \to \Phi_{\mu\alpha\rho}(\mathcal{M}) \to \mathcal{N}.$$

Again because of ($\to$-E) it suffices to show

$$\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma s \in \mathcal{M} \to \mathcal{N}.$$

Use ($\to$-I). Let $r \in \mathcal{M}$. We have to show

$$r\mathrm{E}_\mu^+\sigma s \in \mathcal{N}.$$

This follows from the preceding lemma because $r \in \mathcal{M} \subseteq \mu_E(\Phi)$ and $\mu_E(\Phi)$ is a post-fixed-point of Ψ_E. □

Note that we used from $\mu_E(\Phi)$ the properties of being a pre-fixed-point and a post-fixed-point of Ψ_E. Note also that for the purposes of the normalization proof it would have been enough to show that $\mu_E(\Phi)$ is a pre-fixed-point of Ψ_I instead of $\Psi_I^{\supseteq}$, but from the above result we even get the following

Corollary 9.45 $\mu_I(\Phi) \subseteq \mu_E(\Phi)$.

Proof $\mu_I(\Phi)$ is the least pre-fixed-point of $\Psi_I^{\supseteq}$. □

Lemma 9.46 $\mu_I(\Phi)$ is a post-fixed-point of Ψ_E.

Proof Set $\mathcal{M} := \mu_I(\Phi)$. We prove $\mathcal{M} \subseteq \Psi_E(\mathcal{M})$ by extended induction (as defined in the introduction to chapter 4) on $\mathcal{M}$. Let $\mathcal{M}' := \Psi_E(\mathcal{M}) \cap \mathcal{M}$. We have to show

$$\Psi_I^{\supseteq}(\mathcal{M}') \subseteq \Psi_E(\mathcal{M}).$$

Because $\mathsf{cl}_{\mu\alpha\rho}$ is monotone, it suffices to show

$$M_I^{\supseteq}(\mathcal{M}') \subseteq M_E(\mathcal{M}).$$

Let $r \in M_I^{\supseteq}(\mathcal{M}')$, i. e., $r = C_{\mu\alpha\rho}t$ with $t \in \Phi_{\mu\alpha\rho}(\mathcal{M}'')$ for some $\mathcal{M}'' \in \mathsf{SAT}_{\mu\alpha\rho}$ with $\mathcal{M}'' \subseteq \mathcal{M}'$. We have to show that

$$C_{\mu\alpha\rho}t \in M_E(\mathcal{M}).$$

Let σ be a type, $\mathcal{N} \in \mathsf{SAT}_\sigma$ and $s \in \forall\mathcal{P}.(\mathcal{P} \to \mathcal{M}) \to (\mathcal{P} \to \mathcal{N}) \to \Phi(\mathcal{P}) \to \mathcal{N}$. We have to show that

$$C_{\mu\alpha\rho}t\mathrm{E}_\mu^+\sigma s \in \mathcal{N}.$$

Because $\mathcal{N}$ is saturated it suffices to show

$$s(\mu\alpha\rho)(\lambda y^{\mu\alpha\rho}y)(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma s)t \in \mathcal{N}.$$

Because of ($\to$-E) for saturated sets (in the sequel the named rules will always be those for saturated sets) and $t \in \Phi_{\mu\alpha\rho}(\mathcal{M}'')$ it suffices to show

$$s(\mu\alpha\rho)(\lambda y^{\mu\alpha\rho}y)(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma s) \in \Phi_{\mu\alpha\rho}(\mathcal{M}'') \to \mathcal{N}.$$

Because of ($\forall$-E) we have

$$s(\mu\alpha\rho) \in (\mathcal{M}'' \to \mathcal{M}) \to (\mathcal{M}'' \to \mathcal{N}) \to \Phi_{\mu\alpha\rho}(\mathcal{M}'') \to \mathcal{N}.$$

By help of ($\to$-I) and because $\mathcal{M}'' \subseteq \mathcal{M}$ we know $\lambda y^{\mu\alpha\rho}y \in \mathcal{M}'' \to \mathcal{M}$. Hence by ($\to$-E)

$$s(\mu\alpha\rho)(\lambda y^{\mu\alpha\rho}y) \in (\mathcal{M}'' \to \mathcal{N}) \to \Phi_{\mu\alpha\rho}(\mathcal{M}'') \to \mathcal{N}.$$

Again because of ($\to$-E) it suffices to show

$$\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma s \in \mathcal{M}'' \to \mathcal{N}.$$

Use ($\to$-I). Let $r' \in \mathcal{M}''$. We have to show

$$r'\mathrm{E}_\mu^+\sigma s \in \mathcal{N}.$$

This follows from $r' \in \mathcal{M}'' \subseteq \mathcal{M}' \subseteq \Psi_E(\mathcal{M}) \subseteq M_E(\mathcal{M})$. □

Note that we only used that $\mu_I(\Phi)$ is a least pre-fixed-point. Note also that the stronger induction principle is essential to the argument because we have to show that $\lambda y^{\mu\alpha\rho}y \in \mathcal{M}'' \to \mathcal{M}$. Finally note that some sort of induction is indispensable because otherwise one could never use the rule (β_μ^+) for saturated sets.
If we took the greatest fixed point in the definition of $\mu_E(\Phi)$ (which would be the greatest post-fixed-point by dualization of Tarski's theorem) we would have $\mu_I(\Phi) \subseteq \mu_E(\Phi)$ again as a consequence of the preceding lemma.
Putting everything together we get the following rules for reasoning with $\mu_I(\Phi)$ and $\mu_E(\Phi)$ where we simply write $\mu(\Phi)$ for both of them:

(SAT) $\mu(\Phi) \in \mathsf{SAT}_{\mu\alpha\rho}$.

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

(μ-E$^+$) If $r \in \mu(\Phi)$, σ type, $\mathcal{N} \in \mathsf{SAT}_\sigma$ and $s \in \forall\mathcal{P}.(\mathcal{P} \to \mu(\Phi)) \to (\mathcal{P} \to \mathcal{N}) \to \Phi(\mathcal{P}) \to \mathcal{N}$, then $r\mathrm{E}_\mu^+\sigma s \in \mathcal{N}$.

Computability predicates for MeIT–it

Extend the definition of $\mathsf{SC}_\rho[\vec{\alpha} := \vec{\mathcal{P}}]$ by the following clause:

(μ) $\mathsf{SC}_{\mu\alpha\rho}[\vec{\alpha} := \vec{\mathcal{P}}] := \mu(\Phi)$ with $\Phi = (\Phi_\sigma)_\sigma$, where $\Phi_\sigma : \mathsf{SAT}_\sigma \to \mathsf{SAT}_{\rho[\vec{\alpha},\alpha:=\vec{\sigma},\sigma]}$ is defined by $\Phi_\sigma(\mathcal{P}) := \mathsf{SC}_\rho[\vec{\alpha},\alpha := \vec{\mathcal{P}},\mathcal{P}]$ (with the same assumptions as in the definition of strong computability for universal types in order to guarantee type-correctness).

The definition will be written more intuitively as follows:

$$\mathsf{SC}_{\mu\alpha\rho}[\vec{\alpha} := \vec{\mathcal{P}}] := \mu\mathcal{P}.\mathsf{SC}_\rho[\vec{\alpha},\alpha := \vec{\mathcal{P}},\mathcal{P}].$$

We want to extend the coincidence and substitution lemma (Lemma 9.13 and Lemma 9.14) to MeIT–it. This is not possible if we interpret "any fixed point of Ψ_E" in the definition of $\mu_E(\Phi)$[18] too liberally. We have to stipulate that this choice only depends on Ψ_E. This means that we assume a function giving for any monotone $\Psi : \mathsf{SAT}_{\mu\alpha\rho} \to \mathsf{SAT}_{\mu\alpha\rho}$ a fixed point of Ψ and that this function is used for choosing the fixed point in the definition of $\mu_E(\Phi)$[19]. With this extra condition it is obvious that the coincidence and substitution lemma extend to MeIT–it.
Now we extend Lemma 9.16 to MeIT–it:

Lemma 9.47 Let r^ρ be a term, $\vec{x}^{\vec{\rho}}$ a list of variables and $\vec{s}$ an equally long list of terms such that $\vec{s} \in \mathsf{SC}_{\vec{\rho}}[\vec{\alpha} := \vec{\mathcal{P}}^{\vec{\sigma}}]$ for some $\vec{\mathcal{P}} \subseteq \mathsf{SAT}$. Assume that

$$\forall x\forall\pi\forall\pi'.(x^\pi, x^{\pi'} \in \mathsf{FV}(r) \cup \vec{x}^{\vec{\rho}}) \wedge (\pi[\vec{\alpha} := \vec{\sigma}] = \pi'[\vec{\alpha} := \vec{\sigma}]) \Rightarrow \pi = \pi'.$$

Then $r[\vec{\alpha} := \vec{\sigma}][\vec{x}^{\vec{\rho}[\vec{\alpha}:=\vec{\sigma}]} := \vec{s}] \in \mathsf{SC}_\rho[\vec{\alpha} := \vec{\mathcal{P}}]$.

Proof Induction on r. Again abbreviate $r[\vec{\alpha} := \vec{\sigma}][\vec{x}^{\vec{\rho}[\vec{\alpha}:=\vec{\sigma}]} := \vec{s}]$ by r' (for any term r). The assumption in the statement will again be referenced to as "injectivity". We only have to consider the rules (μ-I) and (μ-E$^+$).
(μ-I) Case $C_{\mu\alpha\rho}t$. $(C_{\mu\alpha\rho}t)' = C_{\mu\alpha\rho[\vec{\alpha}:=\vec{\sigma}]}t'$. We may assume that $\alpha \notin \vec{\alpha}$. We have to show $C_{\mu\alpha\rho[\vec{\alpha}:=\vec{\sigma}]}t' \in \mathsf{SC}_{\mu\alpha\rho}[\vec{\alpha} := \vec{\mathcal{P}}]$. By the rule ($\mu$-I) for saturated sets it suffices to show

$$t' \in \mathsf{SC}_\rho[\vec{\alpha},\alpha := \vec{\mathcal{P}},\mathsf{SC}_{\mu\alpha\rho}[\vec{\alpha} := \vec{\mathcal{P}}]].$$

[18] For the restrictions of the choice between μ_I and μ_E see the discussion immediately before Lemma 9.13 on p. 118.
[19] This uniformity assumption will be tacitly made also for the other systems of inductive types.

Because $\mathsf{FV}(C_{\mu\alpha\rho}t) = \mathsf{FV}(t)$, injectivity holds trivially also for t. By induction hypothesis $t' \in \mathsf{SC}_{\rho[\alpha:=\mu\alpha\rho]}[\vec{\alpha} := \vec{\mathcal{P}}]$. By the substitution lemma (which for system F is Lemma 9.14) we have

$$\mathsf{SC}_{\rho[\alpha:=\mu\alpha\rho]}[\vec{\alpha} := \vec{\mathcal{P}}] = \mathsf{SC}_{\rho}[\vec{\alpha}, \alpha := \vec{\mathcal{P}}, \mathsf{SC}_{\mu\alpha\rho}[\vec{\alpha} := \vec{\mathcal{P}}]].$$

(μ-E$^+$) Case $r\mathrm{E}^+_\mu \sigma s$. $(r\mathrm{E}^+_\mu \sigma s)' = r'\mathrm{E}^+_\mu(\sigma[\vec{\alpha} := \vec{\sigma}])s'$. We have to show that $r'\mathrm{E}^+_\mu\sigma[\vec{\alpha} := \vec{\sigma}]s' \in \mathsf{SC}_\sigma[\vec{\alpha} := \vec{\mathcal{P}}]$. Use ($\mu$-E$^+$) for saturated sets. Show that $r' \in \mathsf{SC}_{\mu\alpha\rho}[\vec{\alpha} := \vec{\mathcal{P}}]$ and

$$s' \in \forall\mathcal{P}.(\mathcal{P} \to \mathsf{SC}_{\mu\alpha\rho}[\vec{\alpha} := \vec{\mathcal{P}}]) \to (\mathcal{P} \to \mathsf{SC}_\sigma[\vec{\alpha} := \vec{\mathcal{P}}]) \to \mathsf{SC}_\rho[\vec{\alpha}, \alpha := \vec{\mathcal{P}}, \mathcal{P}] \to \mathsf{SC}_\sigma[\vec{\alpha} := \vec{\mathcal{P}}].$$

Injectivity holds for r and s because $\mathsf{FV}(r), \mathsf{FV}(s) \subseteq \mathsf{FV}(r\mathrm{E}^+_\mu \sigma s)$. Therefore by induction hypothesis $r' \in \mathsf{SC}_{\mu\alpha\rho}[\vec{\alpha} := \vec{\mathcal{P}}]$ and $s' \in \mathsf{SC}_{\forall\alpha.(\alpha\to\mu\alpha\rho)\to(\alpha\to\sigma)\to\rho\to\sigma}[\vec{\alpha} := \vec{\mathcal{P}}]$. Unwinding the definition and using the coincidence lemma ($\alpha \notin \mathsf{FV}(\mu\alpha\rho) \cup \mathsf{FV}(\sigma)$) and $\alpha \notin \vec{\alpha}$ (which we assumed in the definition of $\mathsf{SC}_{\mu\alpha\rho}[\vec{\alpha} := \vec{\mathcal{P}}]$) leads to the desired result. □

As for system eF we get as corollaries that $\mathsf{SN} = \Lambda$, that every term is strongly normalizing, that Ω is a normalizing function and that there is no closed term of type 0[20].

9.4.2 Strong normalization of coMeIT–it

Lemma 9.48 If $t \in \mathsf{sn}$, then $C_{\mu\alpha\rho}t \in \mathsf{sn}$.

Proof Induction on $t \in \mathsf{sn}$. □

Lemma 9.49 If $s\Big(\Lambda\alpha\lambda z^{\mu\alpha\rho\times\sigma\to\alpha}.t\alpha\Big(\lambda x^{\mu\alpha\rho}.z\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma s)x\rangle\Big)\Big)\vec{s} \in \mathsf{sn}$, the side conditions of the (β^+_μ)-rule are met and $(C_{\mu\alpha\rho}t)\mathrm{E}^+_\mu\sigma s\vec{s}$ is a term, then $(C_{\mu\alpha\rho}t)\mathrm{E}^+_\mu\sigma s\vec{s} \in \mathsf{sn}$.

Proof Induction on $s\Big(\Lambda\alpha\lambda z^{\mu\alpha\rho\times\sigma\to\alpha}.t\alpha\Big(\lambda x^{\mu\alpha\rho}.z\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma s)x\rangle\Big)\Big)\vec{s} \in \mathsf{sn}^+$ by the usual analysis of the reducts of $(C_{\mu\alpha\rho}t)\mathrm{E}^+_\mu\sigma s\vec{s}$. (Note that we need sn^+ because s occurs twice in the canonical reduct of $(C_{\mu\alpha\rho}t)\mathrm{E}^+_\mu\sigma s\vec{s}$.) □

Corollary 9.50 $\mathsf{SN} \subseteq \mathsf{sn}$.

Proof The same as for Lemma 9.6. □

Saturated sets for coMeIT–it

Extend the definition for system eF of $\mathcal{M}$ being a τ-saturated set by the following clause:

(β^+_μ) If $s\Big(\Lambda\alpha\lambda z^{\mu\alpha\rho\times\sigma\to\alpha}.t\alpha\Big(\lambda x^{\mu\alpha\rho}.z\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma s)x\rangle\Big)\Big)\vec{s} \in \mathcal{M}$, the side conditions of the (β^+_μ)-rule are met and $(C_{\mu\alpha\rho}t)\mathrm{E}^+_\mu\sigma s\vec{s}$ is a term, then $(C_{\mu\alpha\rho}t)\mathrm{E}^+_\mu\sigma s\vec{s} \in \mathcal{M}$.

Extend the definition of the τ-saturated closure $\mathsf{cl}_\tau(M)$ accordingly by the clause

(β^+_μ) If $s\Big(\Lambda\alpha\lambda z^{\mu\alpha\rho\times\sigma\to\alpha}.t\alpha\Big(\lambda x^{\mu\alpha\rho}.z\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma s)x\rangle\Big)\Big)\vec{s} \in \mathsf{cl}_\tau(M)$, the side conditions of the (β^+_μ)-rule are met and $(C_{\mu\alpha\rho}t)\mathrm{E}^+_\mu\sigma s\vec{s}$ is a term, then $(C_{\mu\alpha\rho}t)\mathrm{E}^+_\mu\sigma s\vec{s} \in \mathsf{cl}_\tau(M)$.

Again $\mathsf{cl}_\tau(M)$ is the smallest τ-saturated set containing $M \cap \mathsf{SN}_\tau$.

[20] As was mentioned in footnote 13 concerning system F we never used induction on SN. This is also true of this system (and of all the others considered in this thesis). It would not be possible e.g. in the following variant of MeIT–it with (μ-E$^+$) replaced by: If $r : \mu\alpha\rho$ and σ type, then $r\mathrm{E}^+_\mu\sigma : (\forall\alpha.(\alpha \to \mu\alpha\rho) \to (\alpha \to \sigma) \to \rho \to \sigma) \to \sigma$. We would have to prove $r^{\mu\alpha\rho} \in \mathsf{SN} \Rightarrow r\mathrm{E}^+_\mu\sigma \in \mathsf{SN}$ by induction on SN.

Calculating with saturated sets for coMeIT–it

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

$$\begin{array}{rcl} M_I(\mathcal{M}) & := & \{r \in \Lambda_{\mu\alpha\rho} | \exists t \in \Big(\forall\mathcal{P}.(\mathcal{M} \to \mathcal{P}) \to \Phi(\mathcal{P})\Big).r = C_{\mu\alpha\rho}t\} \\ M_E(\mathcal{M}) & := & \{r \in \Lambda_{\mu\alpha\rho} | \forall\sigma\forall\mathcal{N} \in \mathsf{SAT}_\sigma \forall s \in \Big(\forall\mathcal{P}.(\mathcal{M} \times \mathcal{N} \to \mathcal{P}) \to \Phi(\mathcal{P})\Big) \to \mathcal{N}.r\mathrm{E}^+_\mu s \in \mathcal{N}\} \end{array}$$

and set

$$\mu_I(\Phi) := \text{ the least fixed point of } \Psi_I : \mathsf{SAT}_{\mu\alpha\rho} \to \mathsf{SAT}_{\mu\alpha\rho},$$

defined by

$$\Psi_I(\mathcal{M}) := \mathsf{cl}_{\mu\alpha\rho}(M_I(\mathcal{M}))$$

and

$$\mu_E(\Phi) := \text{ any fixed point of } \Psi_E : \mathsf{SAT}_{\mu\alpha\rho} \to \mathsf{SAT}_{\mu\alpha\rho},$$

defined by

$$\Psi_E(\mathcal{M}) := \mathsf{cl}_{\mu\alpha\rho}(M_E(\mathcal{M})).$$

Because Ψ_I and Ψ_E are monotone (they are even non-strictly positive) and $\mathsf{SAT}_{\mu\alpha\rho}$ is a complete lattice, such fixed points exist. Hence, $\mu_I(\Phi)$ and $\mu_E(\Phi)$ are $\mu\alpha\rho$-saturated sets. $\mu_I(\Phi)$ is (pointwise) isotone in the argument Φ. If we took the least fixed point for $\mu_E(\Phi)$, the same would hold.
The first approach corresponds to the encoding

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

for $\beta \notin \{\alpha\} \cup \mathsf{FV}(\rho)$. Hence $(\mu\alpha\rho)' = \mathsf{pos}(\mu\alpha\rho')$ which is the encoding used for embedding coMeIT into coMePIT in section 6.3.3.
The second approach (via $\mu_E(\Phi)$) corresponds to

$$(\mu\alpha\rho)' := \hat{\mu}\beta\forall\gamma.((\forall\alpha.(\beta \times \gamma \to \alpha) \to \rho') \to \gamma) \to \gamma$$

with "new" β and γ. Following the proof below one even gets an embedding into a system of non-interleaved positive fixed-point types (see appendix B).

Lemma 9.51 $M_E(\mathcal{M}) \cap \mathsf{SN} \in \mathsf{SAT}$ (which implies $\Psi_E(\mathcal{M}) = M_E(\mathcal{M}) \cap \mathsf{SN}$).

Proof Straightforward. □

Lemma 9.52 $M_I(\mathcal{M}) \subseteq \mathsf{SN}$ (which implies $\Psi_I(\mathcal{M}) \supseteq M_I(\mathcal{M})$). □

Lemma 9.53 A $\mu\alpha\rho$-saturated set $\mathcal{M}$ is a pre-fixed-point of Ψ_I iff for all $t \in \forall\mathcal{P}.(\mathcal{M} \to \mathcal{P}) \to \Phi(\mathcal{P})$, we have that $C_{\mu\alpha\rho}t \in \mathcal{M}$. A $\mu\alpha\rho$-saturated set $\mathcal{M}$ is a post-fixed-point of Ψ_E iff for all $r \in \mathcal{M}$, types σ, $\mathcal{N} \in \mathsf{SAT}_\sigma$ and $s \in \Big(\forall\mathcal{P}.(\mathcal{M} \times \mathcal{N} \to \mathcal{P}) \to \Phi(\mathcal{P})\Big) \to \mathcal{N}$, we have $r\mathrm{E}^+_\mu\sigma s \in \mathcal{N}$.

Proof Use the two preceding lemmas. □

Lemma 9.54 $\mu_E(\Phi)$ is a pre-fixed-point of Ψ_I.

Proof Set $\mathcal{M} := \mu_E(\Phi)$. We have to show that

$$\Psi_I(\mathcal{M}) \subseteq \mathcal{M}.$$

Because $\mathcal{M}$ is a pre-fixed-point of Ψ_E, it suffices to show

$$\Psi_I(\mathcal{M}) \subseteq \Psi_E(\mathcal{M}).$$

Because $\mathsf{cl}_{\mu\alpha\rho}$ is monotone it suffices to show

$$M_I(\mathcal{M}) \subseteq M_E(\mathcal{M}).$$

Let $t \in \forall\mathcal{P}.(\mathcal{M} \to \mathcal{P}) \to \Phi(\mathcal{P})$. We have to show

$$C_{\mu\alpha\rho}t \in M_E(\mathcal{M}).$$

Let σ be a type, $\mathcal{N} \in \mathsf{SAT}_\sigma$ and $s \in \Big(\forall\mathcal{P}.(\mathcal{M} \times \mathcal{N} \to \mathcal{P}) \to \Phi(\mathcal{P})\Big) \to \mathcal{N}$. We have to show that

$$C_{\mu\alpha\rho}t\mathrm{E}^+_\mu\sigma s \in \mathcal{N}.$$

Because $\mathcal{N}$ is saturated it suffices to show

$$s\Big(\Lambda\alpha\lambda z^{\mu\alpha\rho\times\sigma\to\alpha}.t\alpha\Big(\lambda x^{\mu\alpha\rho}.z\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma s)x\rangle\Big)\Big) \in \mathcal{N}.$$

Because of ($\to$-E) for saturated sets (in the sequel the named rules will always be those for saturated sets) it suffices to show

$$\Lambda\alpha\lambda z^{\mu\alpha\rho\times\sigma\to\alpha}.t\alpha\Big(\lambda x^{\mu\alpha\rho}.z\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma s)x\rangle\Big) \in \forall\mathcal{P}.(\mathcal{M} \times \mathcal{N} \to \mathcal{P}) \to \Phi(\mathcal{P}).$$

We use ($\forall$-I). Let τ be a type and $\mathcal{P} \in \mathsf{SAT}_\tau$. We have to show

$$\lambda z^{\mu\alpha\rho\times\sigma\to\tau}.t\tau\Big(\lambda x^{\mu\alpha\rho}.z\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma s)x\rangle\Big) \in (\mathcal{M} \times \mathcal{N} \to \mathcal{P}) \to \Phi_\tau(\mathcal{P}).$$

We use ($\to$-I). Let $u \in \mathcal{M} \times \mathcal{N} \to \mathcal{P}$. We may assume that $x^{\mu\alpha\rho} \notin \mathsf{FV}(u)$. Therefore we have to show

$$t\tau\Big(\lambda x^{\mu\alpha\rho}.u\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma s)x\rangle\Big) \in \Phi_\tau(\mathcal{P}).$$

Because of ($\forall$-E) we have that $t\tau \in (\mathcal{M} \to \mathcal{P}) \to \Phi_\tau(\mathcal{P})$. By ($\to$-E) it suffices to show

$$\lambda x^{\mu\alpha\rho}.u\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma s)x\rangle \in \mathcal{M} \to \mathcal{P}.$$

We use ($\to$-I). Let $r \in \mathcal{M}$. We have to show

$$u\langle r, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma s)r\rangle \in \mathcal{P}.$$

Because of ($\to$-E) and ($\times$-I) it suffices to show

$$r \in \mathcal{M} \text{ and } (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma s)r \in \mathcal{N}.$$

Because $r \in \mathcal{M} \subseteq \mathsf{SN}$ and $\mathcal{N}$ is saturated it suffices to show

$$r\mathrm{E}^+_\mu\sigma s \in \mathcal{N}.$$

This follows from the preceding lemma because $\mathcal{M}$ is a post-fixed-point of Ψ_E. □

Note that we used from $\mu_E(\Phi)$ the properties of being a pre-fixed-point and a post-fixed-point of Ψ_E.

Corollary 9.55 $\mu_I(\Phi) \subseteq \mu_E(\Phi)$.

Proof $\mu_I(\Phi)$ is the least pre-fixed-point of Ψ_I. □

Lemma 9.56 $\mu_I(\Phi)$ is a post-fixed-point of Ψ_E.

Proof Set $\mathcal{M} := \mu_I(\Phi)$. We prove $\mathcal{M} \subseteq \Psi_E(\mathcal{M})$ by extended induction (as defined in the introduction to chapter 4) on $\mathcal{M}$. Let $\mathcal{M}' := \Psi_E(\mathcal{M}) \cap \mathcal{M}$. We have to show

$$\Psi_I(\mathcal{M}') \subseteq \Psi_E(\mathcal{M}).$$

Because $\mathsf{cl}_{\mu\alpha\rho}$ is monotone it suffices to show

$$M_I(\mathcal{M}') \subseteq M_E(\mathcal{M}).$$

Let $r \in M_I(\mathcal{M}')$, i. e., $r = C_{\mu\alpha\rho}t$ with $t \in \forall\mathcal{P}.(\mathcal{M}' \to \mathcal{P}) \to \mathcal{N}$. We have to show that

$$C_{\mu\alpha\rho}t \in M_E(\mathcal{M}).$$

Let σ be a type, $\mathcal{N} \in \mathsf{SAT}_\sigma$ and $s \in \Big(\forall\mathcal{P}.(\mathcal{M} \times \mathcal{N} \to \mathcal{P}) \to \Phi(\mathcal{P})\Big) \to \mathcal{N}$. We have to show

$$C_{\mu\alpha\rho}t\mathrm{E}^+_\mu\sigma s \in \mathcal{N}.$$

Because $\mathcal{N}$ is saturated it suffices to show

$$s\Big(\Lambda\alpha\lambda z^{\mu\alpha\rho\times\sigma\to\alpha}.t\alpha\Big(\lambda x^{\mu\alpha\rho}.z\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma s)x\rangle\Big)\Big) \in \mathcal{N}.$$

Because of ($\to$-E) it suffices to show

$$\Lambda\alpha\lambda z^{\mu\alpha\rho\times\sigma\to\alpha}.t\alpha\Big(\lambda x^{\mu\alpha\rho}.z\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma s)x\rangle\Big) \in \forall\mathcal{P}.(\mathcal{M} \times \mathcal{N} \to \mathcal{P}) \to \Phi(\mathcal{P}).$$

We use ($\forall$-I). Let τ be a type and $\mathcal{P} \in \mathsf{SAT}_\tau$. We have to show

$$\lambda z^{\mu\alpha\rho\times\sigma\to\tau}.t\tau\Big(\lambda x^{\mu\alpha\rho}.z\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma s)x\rangle\Big) \in (\mathcal{M} \times \mathcal{N} \to \mathcal{P}) \to \Phi_\tau(\mathcal{P}).$$

We use ($\to$-I). Let $u \in \mathcal{M} \times \mathcal{N} \to \mathcal{P}$. We may assume that $x^{\mu\alpha\rho} \notin \mathsf{FV}(u)$. Therefore we have to show

$$t\tau\Big(\lambda x^{\mu\alpha\rho}.u\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma s)x\rangle\Big) \in \Phi_\tau(\mathcal{P}).$$

Because of ($\forall$-E) we have that $t\tau \in (\mathcal{M}' \to \mathcal{P}) \to \Phi_\tau(\mathcal{P})$. By ($\to$-E) it suffices to show

$$\lambda x^{\mu\alpha\rho}.u\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma s)x\rangle \in \mathcal{M}' \to \mathcal{P}.$$

We use ($\to$-I). Let $r' \in \mathcal{M}'$. We have to show

$$u\langle r', (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma s)r'\rangle \in \mathcal{P}.$$

Because of ($\to$-E) and ($\times$-I) it suffices to show

$$r' \in \mathcal{M} \text{ and } (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma s)r' \in \mathcal{N}.$$

Because $r' \in \mathcal{M}' \subseteq \mathcal{M} \subseteq \mathsf{SN}$ and $\mathcal{N}$ is saturated it suffices to show

$$r'\mathrm{E}^+_\mu\sigma s \in \mathcal{N}.$$

This follows from $r' \in \mathcal{M}' \subseteq \Psi_E(\mathcal{M}) \subseteq M_E(\mathcal{M})$. □

Note that we only used that $\mu_I(\Phi)$ is a least pre-fixed-point. Note also that the stronger induction principle is essential to the argument because we have to show that $r' \in \mathcal{M}$. Finally note that some sort of induction is indispensable because otherwise one could never use the rule (β^+_μ) for saturated sets.
If we took the greatest fixed point in the definition of $\mu_E(\Phi)$ we would have $\mu_I(\Phi) \subseteq \mu_E(\Phi)$ again as a consequence of the preceding lemma.
Putting everything together we get the following rules for reasoning with $\mu_I(\Phi)$ and $\mu_E(\Phi)$ where we simply write $\mu(\Phi)$ for both of them:

(SAT) $\mu(\Phi) \in \mathsf{SAT}_{\mu\alpha\rho}$.

(μ-I) If $t \in \forall\mathcal{P}.(\mu(\Phi) \to \mathcal{P}) \to \Phi(\mathcal{P})$, then $C_{\mu\alpha\rho}t \in \mu(\Phi)$.

(μ-E$^+$) If $r \in \mu(\Phi)$, σ type, $\mathcal{N} \in \mathsf{SAT}_\sigma$ and $s \in \Big(\forall\mathcal{P}.(\mu(\Phi) \times \mathcal{N} \to \mathcal{P}) \to \Phi(\mathcal{P})\Big) \to \mathcal{N}$, then $r\mathrm{E}^+_\mu\sigma s \in \mathcal{N}$.

Computability predicates for coMeIT–it

Extend the definition of $\mathsf{SC}_\rho[\vec{\alpha} := \vec{\mathcal{P}}]$ by the following clause:

(μ) $\mathsf{SC}_{\mu\alpha\rho}[\vec{\alpha} := \vec{\mathcal{P}}] := \mu(\Phi)$ with $\Phi = (\Phi_\sigma)_\sigma$, where $\Phi_\sigma : \mathsf{SAT}_\sigma \to \mathsf{SAT}_{\rho[\vec{\alpha},\alpha:=\vec{\sigma},\sigma]}$ is defined by $\Phi_\sigma(\mathcal{P}) := \mathsf{SC}_\rho[\vec{\alpha}, \alpha := \vec{\mathcal{P}}, \mathcal{P}]$ (with the same assumptions as in the definition of strong computability for universal types in order to guarantee type-correctness).

The definition will be written more intuitively as follows:

$$\mathsf{SC}_{\mu\alpha\rho}[\vec{\alpha} := \vec{\mathcal{P}}] := \mu\mathcal{P}.\mathsf{SC}_\rho[\vec{\alpha}, \alpha := \vec{\mathcal{P}}, \mathcal{P}].$$

It is obvious that coincidence and substitution lemma (Lemma 9.13 and Lemma 9.14) extend to coMeIT–it[21].
Now we extend Lemma 9.16 to coMeIT–it:

Lemma 9.57 Let r^ρ be a term, $\vec{x}^{\vec{\rho}}$ a list of variables and $\vec{s}$ an equally long list of terms such that $\vec{s} \in \mathsf{SC}_{\vec{\rho}}[\vec{\alpha} := \vec{\mathcal{P}}^{\vec{\sigma}}]$ for some $\vec{\mathcal{P}} \subseteq \mathsf{SAT}$. Assume that

$$\forall x \forall \pi \forall \pi'.(x^\pi, x^{\pi'} \in \mathsf{FV}(r) \cup \vec{x}^{\vec{\rho}}) \wedge (\pi[\vec{\alpha} := \vec{\sigma}] = \pi'[\vec{\alpha} := \vec{\sigma}]) \Rightarrow \pi = \pi'.$$

Then $r[\vec{\alpha} := \vec{\sigma}][\vec{x}^{\vec{\rho}[\vec{\alpha}:=\vec{\sigma}]} := \vec{s}] \in \mathsf{SC}_\rho[\vec{\alpha} := \vec{\mathcal{P}}]$.

Proof Induction on r. Again abbreviate $r[\vec{\alpha} := \vec{\sigma}][\vec{x}^{\vec{\rho}[\vec{\alpha}:=\vec{\sigma}]} := \vec{s}]$ by r' (for any term r). The assumption in the statement will again be referenced to as "injectivity". We only have to consider the rules (μ-I) and (μ-E$^+$).
(μ-I) Case $C_{\mu\alpha\rho}t$. $(C_{\mu\alpha\rho}t)' = C_{\mu\alpha\rho[\vec{\alpha}:=\vec{\sigma}]}t'$. We have to show $C_{\mu\alpha\rho[\vec{\alpha}:=\vec{\sigma}]}t' \in \mathsf{SC}_{\mu\alpha\rho}[\vec{\alpha} := \vec{\mathcal{P}}]$. By the rule ($\mu$-I) for saturated sets it suffices to show

$$t' \in \forall\mathcal{P}.(\mathsf{SC}_{\mu\alpha\rho}[\vec{\alpha} := \vec{\mathcal{P}}] \to \mathcal{P}) \to \mathsf{SC}_\rho[\vec{\alpha}, \alpha := \vec{\mathcal{P}}, \mathcal{P}].$$

Because $\mathsf{FV}(C_{\mu\alpha\rho}t) = \mathsf{FV}(t)$, injectivity holds trivially also for t. By induction hypothesis $t' \in \mathsf{SC}_{\forall\alpha.(\mu\alpha\rho\to\alpha)\to\rho}[\vec{\alpha} := \vec{\mathcal{P}}]$. By definition of strong computability,

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

[21] Cf. the discussion for MeIT–it.

Using the implicit assumption $\alpha \notin \vec{\alpha}$, we get $\mathsf{SC}_\alpha[\vec{\alpha}, \alpha := \vec{\mathcal{P}}, \mathcal{P}]) = \mathcal{P}$. By the coincidence lemma (for F this is Lemma 9.13), we get $\mathsf{SC}_{\mu\alpha\rho}[\vec{\alpha}, \alpha := \vec{\mathcal{P}}, \mathcal{P}] = \mathsf{SC}_{\mu\alpha\rho}[\vec{\alpha} := \vec{\mathcal{P}}]$.
(μ-E$^+$) Case $r\mathrm{E}^+_\mu\sigma s$. $(r\mathrm{E}^+_\mu\sigma s)' = r'\mathrm{E}^+_\mu(\sigma[\vec{\alpha} := \vec{\sigma}])s'$. We have to show that $r'\mathrm{E}^+_\mu\sigma[\vec{\alpha} := \vec{\sigma}]s' \in \mathsf{SC}_\sigma[\vec{\alpha} := \vec{\mathcal{P}}]$. Use ($\mu$-E$^+$) for saturated sets. Show that $r' \in \mathsf{SC}_{\mu\alpha\rho}[\vec{\alpha} := \vec{\mathcal{P}}]$ and

$$s' \in \Big(\forall\mathcal{P}.(\mathsf{SC}_{\mu\alpha\rho}[\vec{\alpha} := \vec{\mathcal{P}}] \times \mathsf{SC}_\sigma[\vec{\alpha} := \vec{\mathcal{P}}] \to \mathcal{P}) \to \mathsf{SC}_\rho[\vec{\alpha}, \alpha := \vec{\mathcal{P}}, \mathcal{P}]\Big) \to \mathsf{SC}_\sigma[\vec{\alpha} := \vec{\mathcal{P}}].$$

Injectivity holds for r and s because $\mathsf{FV}(r), \mathsf{FV}(s) \subseteq \mathsf{FV}(r\mathrm{E}^+_\mu\sigma s)$. Therefore by induction hypothesis $r' \in \mathsf{SC}_{\mu\alpha\rho}[\vec{\alpha} := \vec{\mathcal{P}}]$ and $s' \in \mathsf{SC}_{(\forall\alpha.(\mu\alpha\rho\times\sigma\to\alpha)\to\rho)\to\sigma}[\vec{\alpha} := \vec{\mathcal{P}}]$. Unwinding the definition and using the coincidence lemma ($\alpha \notin \mathsf{FV}(\mu\alpha\rho) \cup \mathsf{FV}(\sigma)$) and $\alpha \notin \vec{\alpha}$ leads to the desired result. □

As before we get as corollaries that $\mathsf{SN} = \Lambda$, that every term is strongly normalizing, that Ω is a normalizing function and that there is no closed term of type 0.

9.5 Strong normalization of EMIT, varEMIT, IMIT and varIMIT

Direct proofs of strong normalization are given although strong normalization follows already via the embeddings into MeIT–it and coMeIT–it. Technically it is interesting that monotonicity of the operators is mainly enforced not by the general method described in the introduction to the chapter on systems à la Mendler but by a construction based on the monotonicity witnesses which for the elimination-based proof for EMIT follows an idea of [Alt93c]. Because for the systems varEMIT and varIMIT this construction does partly not work the general method is applied almost purely.

9.5.1 Strong normalization of EMIT

Lemma 9.58 If $m \in \mathsf{sn}$ and $t \in \mathsf{sn}$, then $C_{\mu\alpha\rho}mt \in \mathsf{sn}$.

Proof Induction on $m \in \mathsf{sn}$ and $t \in \mathsf{sn}$. □

Lemma 9.59 If $s\Big(m(\mu\alpha\rho)\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma s)t\Big)\vec{s} \in \mathsf{sn}$, $x^{\mu\alpha\rho} \notin \mathsf{FV}(s)$ and $(C_{\mu\alpha\rho}mt)\mathrm{E}_\mu\sigma s\vec{s}$ is a term, then $(C_{\mu\alpha\rho}mt)\mathrm{E}_\mu\sigma s\vec{s} \in \mathsf{sn}$.

Proof Induction on $s\Big(m(\mu\alpha\rho)\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma s)t\Big)\vec{s} \in \mathsf{sn}^+$ by the usual analysis of the reducts of $(C_{\mu\alpha\rho}mt)\mathrm{E}_\mu\sigma s\vec{s}$. (Note that we need sn^+ because s occurs twice in the canonical reduct of $(C_{\mu\alpha\rho}mt)\mathrm{E}_\mu\sigma s\vec{s}$.) □

Lemma 9.60 If $s\Big(m(\mu\alpha\rho)(\mu\alpha\rho \times \sigma)\Big(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma s)x\rangle\Big)t\Big)\vec{s} \in \mathsf{sn}$, $x^{\mu\alpha\rho} \notin \mathsf{FV}(s)$ and $(C_{\mu\alpha\rho}mt)\mathrm{E}^+_\mu\sigma s\vec{s}$ is a term, then $(C_{\mu\alpha\rho}mt)\mathrm{E}^+_\mu\sigma s\vec{s} \in \mathsf{sn}$.

Proof Induction on $s\Big(m(\mu\alpha\rho)(\mu\alpha\rho \times \sigma)\Big(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma s)x\rangle\Big)t\Big)\vec{s} \in \mathsf{sn}^+$ by analysis of the reducts of $(C_{\mu\alpha\rho}mt)\mathrm{E}^+_\mu\sigma s\vec{s}$. (Note that we need sn^+ because s occurs twice in the canonical reduct of $(C_{\mu\alpha\rho}mt)\mathrm{E}^+_\mu\sigma s\vec{s}$.) □

Corollary 9.61 $\mathsf{SN} \subseteq \mathsf{sn}$.

Proof The same as for Lemma 9.6. □

Saturated sets for EMIT

Extend the definition for system eF of $\mathcal{M}$ being a τ-saturated set by the following clause:

(β_μ) If $s\Big(m(\mu\alpha\rho)\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma s)t\Big)\vec{s} \in \mathcal{M}$, $x^{\mu\alpha\rho} \notin \mathsf{FV}(s)$ and $(C_{\mu\alpha\rho}mt)\mathrm{E}_\mu\sigma s\vec{s}$ is a term, then $(C_{\mu\alpha\rho}mt)\mathrm{E}_\mu\sigma s\vec{s} \in \mathcal{M}$.

(β_μ^+) If $s\Big(m(\mu\alpha\rho)(\mu\alpha\rho \times \sigma)\Big(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma s)x\rangle\Big)t\Big)\vec{s} \in \mathcal{M}$, $x^{\mu\alpha\rho} \notin \mathsf{FV}(s)$ and $(C_{\mu\alpha\rho}mt)\mathrm{E}_\mu^+\sigma s\vec{s}$ is a term, then $(C_{\mu\alpha\rho}mt)\mathrm{E}_\mu^+\sigma s\vec{s} \in \mathcal{M}$.

Extend the definition of the τ-saturated closure $\mathsf{cl}_\tau(M)$ accordingly by the clause

(β_μ) If $s\Big(m(\mu\alpha\rho)\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma s)t\Big)\vec{s} \in \mathsf{cl}_\tau(M)$, $x^{\mu\alpha\rho} \notin \mathsf{FV}(s)$ and $(C_{\mu\alpha\rho}mt)\mathrm{E}_\mu\sigma s\vec{s}$ is a term, then $(C_{\mu\alpha\rho}mt)\mathrm{E}_\mu\sigma s\vec{s} \in \mathsf{cl}_\tau(M)$.

(β_μ^+) If $s\Big(m(\mu\alpha\rho)(\mu\alpha\rho \times \sigma)\Big(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma s)x\rangle\Big)t\Big)\vec{s} \in \mathsf{cl}_\tau(M)$, $x^{\mu\alpha\rho} \notin \mathsf{FV}(s)$ and $(C_{\mu\alpha\rho}mt)\mathrm{E}_\mu^+\sigma s\vec{s}$ is a term, then $(C_{\mu\alpha\rho}mt)\mathrm{E}_\mu^+\sigma s\vec{s} \in \mathsf{cl}_\tau(M)$.

Again $\mathsf{cl}_\tau(M)$ is the smallest τ-saturated set containing $M \cap \mathsf{SN}_\tau$.

Calculating with saturated sets for EMIT

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

$$\begin{aligned} M_I(\mathcal{M}) &:= \{r \in \Lambda_{\mu\alpha\rho} | \exists m \in \forall\mathcal{P}\forall\mathcal{Q}.(\mathcal{P} \to \mathcal{Q}) \to \Phi(\mathcal{P}) \to \Phi(\mathcal{Q}) \exists t \in \Phi_{\mu\alpha\rho}(\mathcal{M}).r = C_{\mu\alpha\rho}mt\} \\ M_E(\mathcal{M}) &:= \{r \in \Lambda_{\mu\alpha\rho} | \forall\sigma\forall\mathcal{N} \in \mathsf{SAT}_\sigma.\forall s \in \Phi_\sigma(\mathcal{N}) \to \mathcal{N}\, r\mathrm{E}_\mu\sigma s \in \mathcal{N} \wedge \\ &\qquad \forall s \in \Phi_{\mu\alpha\rho\times\sigma}(\mathcal{M} \times \mathcal{N}) \to \mathcal{N}\, r\mathrm{E}_\mu^+\sigma s \in \mathcal{N}\} \end{aligned}$$

and define $\Psi_I : \mathsf{SAT}_{\mu\alpha\rho} \to \mathsf{SAT}_{\mu\alpha\rho}$ by

$$\Psi_I(\mathcal{M}) := \mathsf{cl}_{\mu\alpha\rho}(M_I(\mathcal{M}))$$

and $\Psi_E : \mathsf{SAT}_{\mu\alpha\rho} \to \mathsf{SAT}_{\mu\alpha\rho}$ by

$$\Psi_E(\mathcal{M}) := \mathsf{cl}_{\mu\alpha\rho}(M_E(\mathcal{M})).$$

We cannot ensure Ψ_I or Ψ_E to be monotone. In order to overcome this problem we define (with Pow denoting the power set) for $m \in \forall\mathcal{P}\forall\mathcal{Q}.(\mathcal{P} \to \mathcal{Q}) \to \Phi(\mathcal{P}) \to \Phi(\mathcal{Q})$

$$\Phi_m^{\supseteq} : \mathsf{SAT}_{\mu\alpha\rho} \to \mathsf{Pow}(\mathsf{SN}_{\rho[\alpha:=\mu\alpha\rho]})$$

via

$$\Phi_m^{\supseteq}(\mathcal{M}) \;:=\; \{t \in \mathsf{SN}_{\rho[\alpha:=\mu\alpha\rho]} | \forall\sigma\forall\mathcal{N} \in \mathsf{SAT}_\sigma \forall s \in \mathcal{M} \to \mathcal{N}.m(\mu\alpha\rho)\sigma st \in \Phi_\sigma(\mathcal{N})\}$$

and $\Phi^{\subseteq} := (\Phi_\sigma^{\subseteq})_\sigma$ with

$$\Phi_\sigma^{\subseteq} : \mathsf{SAT}_\sigma \to \mathsf{SAT}_{\rho[\alpha:=\sigma]}$$

via

$$\begin{aligned} \Phi_\sigma^{\subseteq}(\mathcal{N}) \;:=\; & \mathsf{cl}_{\rho[\alpha:=\sigma]}(\{r \in \Lambda_{\rho[\alpha:=\sigma]} | \exists m \in \forall\mathcal{P}\forall\mathcal{Q}.(\mathcal{P} \to \mathcal{Q}) \to \Phi(\mathcal{P}) \to \Phi(\mathcal{Q}) \\ & \exists\mathcal{M} \in \mathsf{SAT}_{\mu\alpha\rho} \exists s \in \mathcal{M} \to \mathcal{N} \exists t \in \Phi_{\mu\alpha\rho}(\mathcal{M}).r = m(\mu\alpha\rho)\sigma st\}) \end{aligned}$$

$\Phi_m^{\supseteq}$ is monotone (even non-strictly positive) and $\Phi_m^{\supseteq} \supseteq \Phi_{\mu\alpha\rho}$ (pointwise) because of the rules ($\forall$-E) and ($\to$-E) for saturated sets. $\Phi_\sigma^{\subseteq}$ is monotone (even strictly positive) and $\Phi_\sigma^{\subseteq} \subseteq \Phi_\sigma$ (pointwise) also because of the rules ($\forall$-E) and ($\to$-E) for saturated sets. (Note that already the set which is the argument to $\mathsf{cl}_{\rho[\alpha:=\sigma]}$ is a subset of $\Phi_\sigma(\mathcal{M})$ and therefore also $\Phi_\sigma^{\subseteq}(\mathcal{M})$ because $\Phi_\sigma(\mathcal{M})$ is saturated.)

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

$$\begin{aligned} M_I^{\supseteq}(\mathcal{M}) &:= \{r \in \Lambda_{\mu\alpha\rho} | \exists m \in \forall\mathcal{P}\forall\mathcal{Q}.(\mathcal{P} \to \mathcal{Q}) \to \Phi(\mathcal{P}) \to \Phi(\mathcal{Q}) \exists t \in \Phi_m^{\supseteq}(\mathcal{M}).r = C_{\mu\alpha\rho}mt\} \\ M_E^{\subseteq}(\mathcal{M}) &:= \{r \in \Lambda_{\mu\alpha\rho} | \forall\sigma\forall\mathcal{N} \in \mathsf{SAT}_\sigma.\forall s \in \Phi_\sigma(\mathcal{N}) \to \mathcal{N}\, r\mathrm{E}_\mu\sigma s \in \mathcal{N} \wedge \\ &\qquad \forall s \in \Phi_{\mu\alpha\rho\times\sigma}^{\subseteq}(\mathcal{M} \times \mathcal{N}) \to \mathcal{N}\, r\mathrm{E}_\mu^+\sigma s \in \mathcal{N}\} \end{aligned}$$

and set

$$\mu_I(\Phi) := \text{ the least fixed point of } \Psi_I^{\supseteq} : \mathsf{SAT}_{\mu\alpha\rho} \to \mathsf{SAT}_{\mu\alpha\rho},$$

defined by

$$\Psi_I^{\supseteq}(\mathcal{M}) := \mathsf{cl}_{\mu\alpha\rho}(M_I^{\supseteq}(\mathcal{M}))$$

and

$$\mu_E(\Phi) := \text{ any fixed point of } \Psi_E^{\subseteq} : \mathsf{SAT}_{\mu\alpha\rho} \to \mathsf{SAT}_{\mu\alpha\rho},$$

defined by

$$\Psi_E^{\subseteq}(\mathcal{M}) := \mathsf{cl}_{\mu\alpha\rho}(M_E^{\subseteq}(\mathcal{M})).$$

Because Ψ_I and Ψ_E are monotone (they are even non-strictly positive) and $\mathsf{SAT}_{\mu\alpha\rho}$ is a complete lattice, such fixed points exists. Hence, $\mu_I(\Phi)$ and $\mu_E(\Phi)$ are $\mu\alpha\rho$-saturated sets.
Let us extract an encoding of types from the definition of $\mu_I(\Phi)$:

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

where we write $\mathsf{Mon}(\lambda\alpha\rho)$ for $\forall\alpha\forall\beta.(\alpha \to \beta) \to \rho \to \rho[\alpha := \beta]$ and assume that $\beta \notin \{\alpha\} \cup \mathsf{FV}(\rho)$. The extraction read off the following proofs indeed gives an embedding of EMIT into NIPIT. Following the elimination-based proof one presumably gets an embedding into a system of non-interleaving positive fixed-point types.

Lemma 9.62 $M_E(\mathcal{M}) \cap \mathsf{SN} \in \mathsf{SAT}$ and $M_E^{\subseteq}(\mathcal{M}) \cap \mathsf{SN} \in \mathsf{SAT}$ (which implies that $\Psi_E(\mathcal{M}) = M_E(\mathcal{M}) \cap \mathsf{SN}$ and $\Psi_E^{\subseteq}(\mathcal{M}) = M_E^{\subseteq}(\mathcal{M}) \cap \mathsf{SN}$).

Proof Straightforward. □

Lemma 9.63 $M_I(\mathcal{M}) \subseteq \mathsf{SN}$ and $M_I^{\supseteq}(\mathcal{M}) \subseteq \mathsf{SN}$ (which implies that $\Psi_I(\mathcal{M}) \supseteq M_I(\mathcal{M})$ and $\Psi_I^{\supseteq}(\mathcal{M}) \supseteq M_I^{\supseteq}(\mathcal{M})$). □

Lemma 9.64 Let $\mathcal{M}$ be a $\mu\alpha\rho$-saturated set $\mathcal{M}$.
$\mathcal{M}$ is a pre-fixed-point of Ψ_I iff for all $m \in \forall\mathcal{P}\forall\mathcal{Q}.(\mathcal{P} \to \mathcal{Q}) \to \Phi(\mathcal{P}) \to \Phi(\mathcal{Q})$ and for all $t \in \Phi_{\mu\alpha\rho}(\mathcal{M})$, we have that $C_{\mu\alpha\rho}mt \in \mathcal{M}$.
$\mathcal{M}$ is a post-fixed-point of Ψ_E iff for all $r \in \mathcal{M}$, types σ, $\mathcal{N} \in \mathsf{SAT}_\sigma$ and $s \in \Phi_\sigma(\mathcal{N}) \to \mathcal{N}$, we have $r\mathrm{E}_\mu\sigma s \in \mathcal{N}$ and for all $s \in \Phi_{\mu\alpha\rho\times\sigma}(\mathcal{M} \times \mathcal{N}) \to \mathcal{N}$, $r\mathrm{E}_\mu^+\sigma s \in \mathcal{N}$.

Proof Use the two preceding lemmas. □

Lemma 9.65 $\mu_I(\Phi)$ is a pre-fixed-point of Ψ_I.

Proof $\mu_I(\Phi)$ is a pre-fixed-point of $\Psi_I^{\supseteq}$ by definition. Hence

$$\mu_I(\Phi) \supseteq \Psi_I^{\supseteq}(\mu_I(\Phi)) \supseteq \Psi_I(\mu_I(\Phi)),$$

which clearly follows from $\Phi_{\overline{m}}^{\supseteq} \supseteq \Phi_{\mu\alpha\rho}$ because $M_I(\mathcal{M})$ and $M_I^{\supseteq}(\mathcal{M})$ have the same definition with exception of the occurrence of $\Phi_{\mu\alpha\rho}$ and $\Phi_{\overline{m}}^{\supseteq}$, respectively, which is at a strictly positive position. $\square$

Lemma 9.66 $\mu_E(\Phi)$ is a post-fixed-point of Ψ_E.

Proof $\mu_E(\Phi)$ is a post-fixed-point of $\Psi_E^{\subseteq}$ by definition. Hence

$$\mu_E(\Phi) \subseteq \Psi_E^{\subseteq}(\mu_E(\Phi)) \subseteq \Psi_E(\mu_E(\Phi)),$$

which clearly follows from $\Phi^{\subseteq} \subseteq \Phi$ (typewise) because $M_E(\mathcal{M})$ and $M_E^{\subseteq}(\mathcal{M})$ have the same definition with exception of the occurrence of $\Phi_{\mu\alpha\rho\times\sigma}$ and $\Phi_{\mu\alpha\rho\times\sigma}^{\subseteq}$, respectively, which is at a non-strictly positive position. $\square$

Lemma 9.67 $\mu_E(\Phi)$ is a pre-fixed-point of Ψ_I[22].

Proof Set $\mathcal{M} := \mu_E(\Phi)$. We have to show

$$\Psi_I(\mathcal{M}) \subseteq \mathcal{M}.$$

Because $\mathcal{M}$ is a pre-fixed-point of $\Psi_E^{\subseteq}$, it suffices to show

$$\Psi_I(\mathcal{M}) \subseteq \Psi_E^{\subseteq}(\mathcal{M}).$$

Because $\mathsf{cl}_{\mu\alpha\rho}$ is monotone, it suffices to show

$$M_I(\mathcal{M}) \subseteq M_E^{\subseteq}(\mathcal{M}).$$

Let $m \in \forall\mathcal{P}\forall\mathcal{Q}.(\mathcal{P}\to\mathcal{Q})\to\Phi(\mathcal{P})\to\Phi(\mathcal{Q})$ and $t \in \Phi_{\mu\alpha\rho}(\mathcal{M})$. We have to show

$$C_{\mu\alpha\rho}mt \in M_E^{\subseteq}(\mathcal{M}).$$

Let σ be a type, $\mathcal{N} \in \mathsf{SAT}_\sigma$ and $s \in \Phi_\sigma(\mathcal{N}) \to \mathcal{N}$. We have to show that

$$C_{\mu\alpha\rho}mt\mathrm{E}_\mu\sigma s \in \mathcal{N}.$$

Because $\mathcal{N}$ is saturated it suffices to show

$$s\Big(m(\mu\alpha\rho)\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma s)t\Big) \in \mathcal{N}.$$

Because of ($\to$-E) for saturated sets (in the sequel the named rules will always be those for saturated sets) it suffices to show

$$m(\mu\alpha\rho)\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma s)t \in \Phi_\sigma(\mathcal{N}).$$

By ($\forall$-E) and ($\to$-E) it suffices to show

$$\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma s \in \mathcal{M}\to\mathcal{N}.$$

[22] It does not seem possible to show that $\mu_E(\Phi)$ is even a pre-fixed-point of $\Psi_I^{\supseteq}$ and hence I do not know whether $\mu_I(\Phi) \subseteq \mu_E(\Phi)$.

We use ($\to$-I). Let $r \in \mathcal{M}$. We have to show

$$r\mathrm{E}_\mu\sigma s \in \mathcal{N}.$$

This follows from $\mathcal{M} \subseteq \Psi_E^{\subseteq}(\mathcal{M}) \subseteq M_E^{\subseteq}(\mathcal{M})$.
We come to the case for which $\Psi_E^{\subseteq}$ was designed, namely $s \in \Phi_{\mu\alpha\rho\times\sigma}^{\subseteq}(\mathcal{M}\times\mathcal{N}) \to \mathcal{N}$. We have to show

$$C_{\mu\alpha\rho}mt\mathrm{E}_\mu^+\sigma s \in \mathcal{N}.$$

By saturatedness of $\mathcal{N}$ it suffices to show

$$s\Big(m(\mu\alpha\rho)(\mu\alpha\rho\times\sigma)\Big(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma s)x\rangle\Big)t\Big) \in \mathcal{N}.$$

Because of ($\to$-E) it suffices to show

$$m(\mu\alpha\rho)(\mu\alpha\rho\times\sigma)\Big(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma s)x\rangle\Big)t \in \Phi_{\mu\alpha\rho\times\sigma}^{\subseteq}(\mathcal{M}\times\mathcal{N}).$$

According to the definition of $\Phi^{\subseteq}$ (the closure in the definition of $\Phi^{\subseteq}$ only makes the set larger) we are done if we show

$$\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma s)x\rangle \in \mathcal{M}\to\mathcal{M}\times\mathcal{N}.$$

We use ($\to$-I). Let $r \in \mathcal{M}$. We have to show

$$\langle r, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma s)r\rangle \in \mathcal{M}\times\mathcal{N}.$$

By ($\times$-I) it suffices to show

$$(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma s)r \in \mathcal{N}.$$

Because $\mathcal{N}$ is saturated, we only need to show

$$r\mathrm{E}_\mu^+\sigma s \in \mathcal{N}.$$

This follows again from $\mathcal{M} \subseteq \Psi_E^{\subseteq}(\mathcal{M}) \subseteq M_E^{\subseteq}(\mathcal{M})$. $\square$

Note that we used from $\mu_E(\Phi)$ the properties of being a pre-fixed-point and a post-fixed-point of $\Psi_E^{\subseteq}$.

Lemma 9.68 $\mu_I(\Phi)$ is a post-fixed-point of Ψ_E[23].

Proof Set $\mathcal{M} := \mu_I(\Phi)$. We prove $\mathcal{M} \subseteq \Psi_E(\mathcal{M})$ by extended induction (as defined in the introduction to chapter 4) on $\mathcal{M}$. Let $\mathcal{M}' := \Psi_E(\mathcal{M}) \cap \mathcal{M}$. We have to show

$$\Psi_I^{\supseteq}(\mathcal{M}') \subseteq \Psi_E(\mathcal{M}).$$

Because $\mathsf{cl}_{\mu\alpha\rho}$ is monotone, it suffices to show

$$M_I^{\supseteq}(\mathcal{M}') \subseteq M_E(\mathcal{M}).$$

Let $r \in M_I^{\supseteq}(\mathcal{M}')$, i.e., $r = C_{\mu\alpha\rho}mt$ with $m \in \forall\mathcal{P}\forall\mathcal{Q}.(\mathcal{P}\to\mathcal{Q})\to\Phi(\mathcal{P})\to\Phi(\mathcal{Q})$ and $t \in \Phi_{\overline{m}}^{\supseteq}(\mathcal{M}')$. We have to show that

$$C_{\mu\alpha\rho}mt \in M_E(\mathcal{M}).$$

[23] It does not seem possible to show that $\mu_I(\Phi)$ is even a post-fixed-point of $\Psi_E^{\subseteq}$ and hence also this lemma does not give an information whether $\mu_I(\Phi) \subseteq \mu_E(\Phi)$ (in case the greatest fixed point of $\Psi_E^{\subseteq}$ is chosen to give $\mu_E(\Phi)$).

Let σ be a type, $\mathcal{N} \in \mathsf{SAT}_\sigma$ and $s \in \Phi_\sigma(\mathcal{N}) \to \mathcal{N}$. We have to show that

$$C_{\mu\alpha\rho} m t \mathrm{E}_\mu \sigma s \in \mathcal{N}.$$

Because $\mathcal{N}$ is saturated it suffices to show

$$s\Big(m(\mu\alpha\rho)\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma s)t\Big) \in \mathcal{N}.$$

Because of ($\to$-E) for saturated sets it suffices to show

$$m(\mu\alpha\rho)\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma s)t \in \Phi_\sigma(\mathcal{N}).$$

By definition[24] of $\Phi^{\supseteq}_{\overline{m}}(\mathcal{M}')$ it suffices to show

$$\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma s \in \mathcal{M}' \to \mathcal{N}.$$

We use ($\to$-I). Let $r' \in \mathcal{M}'$. We have to show

$$r'\mathrm{E}_\mu\sigma s \in \mathcal{N}.$$

This follows from $\mathcal{M}' \subseteq \Psi_E(\mathcal{M}) \subseteq M_E(\mathcal{M})$.
We come to the case which needs the extended form of induction. Let $s \in \Phi_{\mu\alpha\rho\times\sigma}(\mathcal{M}\times\mathcal{N}) \to \mathcal{N}$. We have to show

$$C_{\mu\alpha\rho} m t \mathrm{E}^+_\mu \sigma s \in \mathcal{N}.$$

By saturatedness of $\mathcal{N}$ it suffices to show

$$s\Big(m(\mu\alpha\rho)(\mu\alpha\rho\times\sigma)\Big(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma s)x\rangle\Big)t\Big) \in \mathcal{N}.$$

Because of ($\to$-E) it suffices to show

$$m(\mu\alpha\rho)(\mu\alpha\rho\times\sigma)\Big(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma s)x\rangle\Big)t \in \Phi_{\mu\alpha\rho\times\sigma}(\mathcal{M}\times\mathcal{N}).$$

By definition of $\Phi^{\supseteq}_{\overline{m}}(\mathcal{M}')$ it suffices to show

$$\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma s)x\rangle \in \mathcal{M}' \to \mathcal{M}\times\mathcal{N}.$$

We use ($\to$-I). Let $r' \in \mathcal{M}'$. We have to show

$$\langle r', (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma s)r'\rangle \in \mathcal{M}\times\mathcal{N}.$$

By ($\times$-I) it suffices to show

$$r' \in \mathcal{M} \text{ and } (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma s)r' \in \mathcal{N}.$$

Because $\mathcal{N}$ is saturated and $\mathcal{M}' \subseteq \mathcal{M}$ (this argument needs the extension of induction), we only need to show

$$r'\mathrm{E}^+_\mu\sigma s \in \mathcal{N}.$$

This follows again from $\mathcal{M}' \subseteq \Psi_E(\mathcal{M}) \subseteq M_E(\mathcal{M})$. □

[24] This would not work if in the definition of $\Phi^{\supseteq}_{\overline{m}}(\mathcal{M}')$ the closure operation would have been used to finally give a saturated set. Therefore locally the realm of saturated sets is left. A formalization in a theory of saturated sets is nevertheless possible because we may view the definition of $\Phi^{\supseteq}_{\overline{m}}(\mathcal{M}')$ as an abbreviation like classes are seen as abbreviations in set theory. In contrast to that taking the closure would correspond to set comprehension.

Note that we only used that $\mu_I(\Phi)$ is a least pre-fixed-point.
Putting everything together we get the following rules for reasoning with $\mu_I(\Phi)$ and $\mu_E(\Phi)$ where we simply write $\mu(\Phi)$ for both of them:

- (SAT) $\mu(\Phi) \in \mathsf{SAT}_{\mu\alpha\rho}$.
- (μ-I) If $m \in \forall\mathcal{P}\forall\mathcal{Q}.(\mathcal{P}\to\mathcal{Q}) \to \Phi(\mathcal{P}) \to \Phi(\mathcal{Q})$ and $t \in \Phi_{\mu\alpha\rho}(\mu(\Phi))$, then $C_{\mu\alpha\rho}mt \in \mu(\Phi)$.
- (μ-E) If $r \in \mu(\Phi)$, σ type, $\mathcal{N} \in \mathsf{SAT}_\sigma$ and $s \in \Phi_\sigma(\mathcal{N}) \to \mathcal{N}$, then $r\mathrm{E}_\mu\sigma s \in \mathcal{N}$.
- (μ-E$^+$) If $r \in \mu(\Phi)$, σ type, $\mathcal{N} \in \mathsf{SAT}_\sigma$ and $s \in \Phi_{\mu\alpha\rho\times\sigma}(\mu(\Phi)\times\mathcal{N}) \to \mathcal{N}$, then $r\mathrm{E}^+_\mu\sigma s \in \mathcal{N}$.

Computability predicates for EMIT

Extend (as for MeIT–it and coMeIT–it) the definition of $\mathsf{SC}_\rho[\vec\alpha := \vec{\mathcal{P}}]$ by the following clause:

(μ) $\mathsf{SC}_{\mu\alpha\rho}[\vec\alpha := \vec{\mathcal{P}}] := \mu(\Phi)$ with $\Phi = (\Phi_\sigma)_\sigma$, where $\Phi_\sigma : \mathsf{SAT}_\sigma \to \mathsf{SAT}_{\rho[\vec\alpha,\alpha:=\vec\sigma,\sigma]}$ is defined by $\Phi_\sigma(\mathcal{P}) := \mathsf{SC}_\rho[\vec\alpha,\alpha := \vec{\mathcal{P}},\mathcal{P}]$ (with the same assumptions as in the definition of strong computability for universal types in order to guarantee type-correctness).

The definition will again be written more intuitively as follows:

$$\mathsf{SC}_{\mu\alpha\rho}[\vec\alpha := \vec{\mathcal{P}}] := \mu\mathcal{P}.\mathsf{SC}_\rho[\vec\alpha,\alpha := \vec{\mathcal{P}},\mathcal{P}].$$

It is obvious that coincidence and substitution lemma (Lemma 9.13 and Lemma 9.14) extend to EMIT[25].
Now we extend Lemma 9.16 to EMIT:

Lemma 9.69 Let r^ρ be a term, $\vec{x}^{\vec\rho}$ a list of variables and $\vec s$ an equally long list of terms such that $\vec s \in \mathsf{SC}_{\vec\rho}[\vec\alpha := \vec{\mathcal{P}}^{\vec\sigma}]$ for some $\vec{\mathcal{P}} \subseteq \mathsf{SAT}$. Assume that

$$\forall x\forall\pi\forall\pi'.(x^\pi, x^{\pi'} \in \mathsf{FV}(r) \cup \vec{x}^{\vec\rho}) \wedge (\pi[\vec\alpha := \vec\sigma] = \pi'[\vec\alpha := \vec\sigma]) \Rightarrow \pi = \pi'.$$

Then $r[\vec\alpha := \vec\sigma][\vec{x}^{\vec\rho[\vec\alpha:=\vec\sigma]} := \vec s] \in \mathsf{SC}_\rho[\vec\alpha := \vec{\mathcal{P}}]$.

Proof Induction on r. Again abbreviate $r[\vec\alpha := \vec\sigma][\vec{x}^{\vec\rho[\vec\alpha:=\vec\sigma]} := \vec s]$ by r' (for any term r). The assumption in the statement will again be referenced to as "injectivity". We only have to consider the rules (μ-I), (μ-E) and (μ-E$^+$).
(μ-I) Case $C_{\mu\alpha\rho}mt$. $(C_{\mu\alpha\rho}mt)' = C_{\mu\alpha\rho[\vec\alpha:=\vec\sigma]}m't'$. We may assume that $\alpha \notin \vec\alpha$. We have to show $C_{\mu\alpha\rho[\vec\alpha:=\vec\sigma]}m't' \in \mathsf{SC}_{\mu\alpha\rho}[\vec\alpha := \vec{\mathcal{P}}]$. By the rule ($\mu$-I) for saturated sets it suffices to show

$$m' \in \forall\mathcal{P}\forall\mathcal{Q}.(\mathcal{P}\to\mathcal{Q}) \to \mathsf{SC}_\rho[\vec\alpha,\alpha := \vec{\mathcal{P}},\mathcal{P}] \to \mathsf{SC}_\rho[\vec\alpha,\alpha := \vec{\mathcal{P}},\mathcal{Q}] \text{ and}$$

$$t' \in \mathsf{SC}_\rho[\vec\alpha,\alpha := \vec{\mathcal{P}}, \mathsf{SC}_{\mu\alpha\rho}[\vec\alpha := \vec{\mathcal{P}}]].$$

Because $\mathsf{FV}(C_{\mu\alpha\rho}mt) = \mathsf{FV}(m) \cup \mathsf{FV}(t)$, injectivity holds trivially also for m and t. By induction hypothesis $m' \in \mathsf{SC}_{\forall\alpha\forall\beta.(\alpha\to\beta)\to\rho\to\rho[\alpha:=\beta]}[\vec\alpha := \vec{\mathcal{P}}]$ and $t' \in \mathsf{SC}_{\rho[\alpha:=\mu\alpha\rho]}[\vec\alpha := \vec{\mathcal{P}}]$. By two applications of the coincidence lemma (which for system F is Lemma 9.13) and the equivalent to Corollary 9.15 (stated for F) we see that

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

[25] See the discussion for MeIT–it.

(one of the applications even needs the generalization which is in fact proved).
By the substitution lemma (which for system F is Lemma 9.14) we have

$$\mathsf{SC}_{\rho[\alpha:=\mu\alpha\rho]}[\vec{\alpha} := \vec{\mathcal{P}}] = \mathsf{SC}_\rho[\vec{\alpha}, \alpha := \vec{\mathcal{P}}, \mathsf{SC}_{\mu\alpha\rho}[\vec{\alpha} := \vec{\mathcal{P}}]].$$

(μ-E) Case $r\mathrm{E}_\mu\sigma s$. This is only a minor variant to the following (μ-E$^+$) and therefore not shown.
(μ-E$^+$) Case $r\mathrm{E}^+_\mu\sigma s$. $(r\mathrm{E}^+_\mu\sigma s)' = r'\mathrm{E}^+_\mu(\sigma[\vec{\alpha} := \vec{\sigma}])s'$. We have to show that

$$r'\mathrm{E}^+_\mu\sigma[\vec{\alpha} := \vec{\sigma}]s' \in \mathsf{SC}_\sigma[\vec{\alpha} := \vec{\mathcal{P}}].$$

Use (μ-E$^+$) for saturated sets. Show that $r' \in \mathsf{SC}_{\mu\alpha\rho}[\vec{\alpha} := \vec{\mathcal{P}}]$ and that

$$s' \in \mathsf{SC}_\rho[\vec{\alpha}, \alpha := \vec{\mathcal{P}}, \mathsf{SC}_{\mu\alpha\rho}[\vec{\alpha} := \vec{\mathcal{P}}] \times \mathsf{SC}_\sigma[\vec{\alpha} := \vec{\mathcal{P}}]] \to \mathsf{SC}_\sigma[\vec{\alpha} := \vec{\mathcal{P}}].$$

Injectivity holds for r and s because $\mathsf{FV}(r), \mathsf{FV}(s) \subseteq \mathsf{FV}(r\mathrm{E}^+_\mu\sigma s)$. Therefore by induction hypothesis $r' \in \mathsf{SC}_{\mu\alpha\rho}[\vec{\alpha} := \vec{\mathcal{P}}]$ and $s' \in \mathsf{SC}_{\rho[\alpha:=\mu\alpha\rho\times\sigma]\to\sigma}[\vec{\alpha} := \vec{\mathcal{P}}]$. Because we may assume $\alpha \notin \vec{\alpha}$ the substitution lemma gives the desired result. □

As before we get as corollaries that $\mathsf{SN} = \Lambda$, that every term is strongly normalizing, that Ω is a normalizing function and that there is no closed term of type 0.
Note that one could have also defined $\Phi^{\subseteq}_{\vec{\sigma}} : \mathsf{SAT}_\sigma \to \mathsf{SAT}_{\rho[\alpha:=\sigma]}$ via

$$\begin{aligned}\Phi^{\subseteq}_{\vec{\sigma}}(\mathcal{N}) \quad := \quad & \mathsf{cl}_{\rho[\alpha:=\sigma]}(\{r \in \Lambda_{\rho[\alpha:=\sigma]} | \exists m \in \forall\mathcal{P}\forall\mathcal{Q}.(\mathcal{P} \to \mathcal{Q}) \to \Phi(\mathcal{P}) \to \Phi(\mathcal{Q}) \\ & \exists\tau\exists\mathcal{M} \in \mathsf{SAT}_\tau\exists s \in \mathcal{M} \to \mathcal{N}\exists t \in \Phi_\tau(\mathcal{M}).r = m\tau\sigma st\})\end{aligned}$$

Every proof concerning $\Phi^{\subseteq}$ would go through without any changes (the new $\Phi^{\subseteq}$ is typewise larger than the former but still smaller than Φ). But the former definition is more informative because it shows that only $\mu\alpha\rho$ is relevant as first argument to m (which is not too surprising).
We also could replace the definition of $\Phi^{\supseteq}_{\vec{m}}(\mathcal{M})$ by a definition of

$$\Phi^{\supseteq}(\mathcal{M}) : \mathsf{SAT}_{\mu\alpha\rho} \to \mathsf{Pow}(\mathsf{SN}_{\rho[\alpha:=\mu\alpha\rho]})$$

(without the parameter m) via

$$\begin{aligned}\Phi^{\supseteq}(\mathcal{M}) \quad := \quad & \{t \in \mathsf{SN}_{\rho[\alpha:=\mu\alpha\rho]} | \forall m \in \forall\mathcal{P}\forall\mathcal{Q}.(\mathcal{P} \to \mathcal{Q}) \to \Phi(\mathcal{P}) \to \Phi(\mathcal{Q}) \\ & \forall\sigma\forall\mathcal{N} \in \mathsf{SAT}_\sigma\forall s \in \mathcal{M} \to \mathcal{N}.m(\mu\alpha\rho)\sigma st \in \Phi_\sigma(\mathcal{N})\}\end{aligned}$$

The proof with $\Phi^{\supseteq}(\mathcal{M})$ instead of $\Phi^{\supseteq}_{\vec{m}}(\mathcal{M})$ would also go through without explicit changes because the universal m may always be instantiated to the existential m in the proof of Lemma 9.68. (Note that $\Phi^{\supseteq}$ is smaller than $\Phi^{\supseteq}_{\vec{m}}$ for every appropriate m but still larger than Φ.)

Acknowledgement: I owe the definition of $\Phi^{\subseteq}$ to [Alt93c] where a version without the closure and with a fixed monotonicity witness and untyped terms in the saturated sets is defined.

9.5.2 Strong normalization of varEMIT

We try to extend the proof for EMIT in order to cover varEMIT. We first have to replace

$$m \in \forall\mathcal{P}\forall\mathcal{Q}.(\mathcal{P} \to \mathcal{Q}) \to \Phi(\mathcal{P}) \to \Phi(\mathcal{Q}) \text{ by } m \in \forall\mathcal{Q}.(\mathcal{M} \to \mathcal{Q}) \to \Phi_{\mu\alpha\rho}(\mathcal{M}) \to \Phi(\mathcal{Q})$$

in the definitions of $M_I(\mathcal{M})$, $\Phi^{\supseteq}(\mathcal{M})$[26], $\Phi^{\subseteq}_{\vec{\sigma}}(\mathcal{N})$ and $M^{\supseteq}_{\vec{I}}(\mathcal{M})$ and remove the first argument to m (the removal of which only type-checks because this argument is always $\mu\alpha\rho$). In the definition

[26] We clearly have to use $\Phi^{\supseteq}(\mathcal{M})$ considered at the end of the proof for EMIT instead of $\Phi^{\supseteq}_{\vec{m}}(\mathcal{M})$.

of $\Phi^{\subseteq}_{\vec{\sigma}}(\mathcal{N})$ we also have to put the existential over m inside the scope of the existential over $\mathcal{M}$. Although these definitions make sense there is no chance to get monotonicity of $\Psi^{\supseteq}_{\vec{I}}$. But $\Psi^{\subseteq}_{\vec{E}}$ is monotone and therefore we may expect the elimination-based proof for EMIT to carry over to varEMIT. That this expectation is justified will be shown in appendix A. Essentially using the general idea of defining $\Psi^{\supseteq}_{\vec{I}}$ as a monotone operator being pointwise larger than Ψ_I presented in the introduction to chapter 6 we also give an introduction-based proof in that appendix. Here only the definition of $\mu_E(\Phi)$ and $\mu_I(\Phi)$ are shown.
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 $\Phi^{\subseteq} := (\Phi^{\subseteq}_{\vec{\sigma}})_\sigma$ with

$$\Phi^{\subseteq}_{\vec{\sigma}} : \mathsf{SAT}_\sigma \to \mathsf{SAT}_{\rho[\alpha:=\sigma]}$$

via

$$\begin{aligned}\Phi^{\subseteq}_{\vec{\sigma}}(\mathcal{N}) \quad := \quad & \mathsf{cl}_{\rho[\alpha:=\sigma]}(\{r \in \Lambda_{\rho[\alpha:=\sigma]} | \exists\mathcal{M} \in \mathsf{SAT}_{\mu\alpha\rho}\exists m \in \forall\mathcal{Q}.(\mathcal{M} \to \mathcal{Q}) \to \Phi_{\mu\alpha\rho}(\mathcal{M}) \to \Phi(\mathcal{Q}) \\ & \exists s \in \mathcal{M} \to \mathcal{N}\exists t \in \Phi_{\mu\alpha\rho}(\mathcal{M}).r = m\sigma st\})\end{aligned}$$

$\Phi^{\subseteq}_{\vec{\sigma}}$ is monotone (even strictly positive) and $\Phi^{\subseteq}_{\vec{\sigma}} \subseteq \Phi_\sigma$ (pointwise) because of the rules ($\forall$-E) and ($\to$-E) for saturated sets.
Define for $\mathcal{M} \in \mathsf{SAT}_{\mu\alpha\rho}$

$$\begin{aligned}M^{\supseteq}_{\vec{I}}(\mathcal{M}) \quad := \quad & \{r \in \Lambda_{\mu\alpha\rho} | \exists\mathcal{M}' \in \mathsf{SAT}_{\mu\alpha\rho}.\mathcal{M}' \subseteq \mathcal{M} \wedge \\ & \exists m \in \forall\mathcal{Q}.(\mathcal{M}' \to \mathcal{Q}) \to \Phi_{\mu\alpha\rho}(\mathcal{M}') \to \Phi(\mathcal{Q})\exists t \in \Phi_{\mu\alpha\rho}(\mathcal{M}').r = C_{\mu\alpha\rho}mt\} \\ M^{\subseteq}_{\vec{E}}(\mathcal{M}) \quad := \quad & \{r \in \Lambda_{\mu\alpha\rho} | \forall\sigma\forall\mathcal{N} \in \mathsf{SAT}_\sigma.\forall s \in \Phi_\sigma(\mathcal{N}) \to \mathcal{N}\, r\mathrm{E}_\mu\sigma s \in \mathcal{N} \wedge \\ & \forall s \in \Phi^{\subseteq}_{\overline{\mu\alpha\rho\times\sigma}}(\mathcal{M} \times \mathcal{N}) \to \mathcal{N}\, r\mathrm{E}^+_\mu\sigma s \in \mathcal{N}\}\end{aligned}$$

and set

$$\mu_I(\Phi) := \text{ the least fixed point of } \Psi^{\supseteq}_{\vec{I}} : \mathsf{SAT}_{\mu\alpha\rho} \to \mathsf{SAT}_{\mu\alpha\rho},$$

defined by

$$\Psi^{\supseteq}_{\vec{I}}(\mathcal{M}) := \mathsf{cl}_{\mu\alpha\rho}(M^{\supseteq}_{\vec{I}}(\mathcal{M}))$$

and set

$$\mu_E(\Phi) := \text{ any fixed point of } \Psi^{\subseteq}_{\vec{E}} : \mathsf{SAT}_{\mu\alpha\rho} \to \mathsf{SAT}_{\mu\alpha\rho},$$

defined by

$$\Psi^{\subseteq}_{\vec{E}}(\mathcal{M}) := \mathsf{cl}_{\mu\alpha\rho}(M^{\subseteq}_{\vec{E}}(\mathcal{M})).$$

Because Ψ_I and Ψ_E are monotone (Ψ_I is even strictly positive and Ψ_E non-strictly positive) and $\mathsf{SAT}_{\mu\alpha\rho}$ is a complete lattice, the fixed points exists. Hence, $\mu_I(\Phi)$ and $\mu_E(\Phi)$ are $\mu\alpha\rho$-saturated sets.
See appendix A for all the details leading to $\mathsf{SN} = \Lambda$ and discussion of the failure of the generalization of the introduction-based proof for EMIT.

9.5.3 Strong normalization of IMIT

We may easily transform the normalization proof for EMIT given in section 9.5.1 to yield a normalization proof for IMIT. We only have to adjust the definition of saturated sets and the closure operation to the different definition of SN and then define the construction of $\mu_I(\Phi)$ and $\mu_E(\Phi)$ as before except that the reference to the monotonicity witness moves from $M_I(\mathcal{M})$ to $M_E(\mathcal{M})$ and from $M^{\supseteq}_{\vec{I}}(\mathcal{M})$ to $M^{\subseteq}_{\vec{E}}(\mathcal{M})$ (the quantifier changes from $\exists$ to $\forall$) and that the parameter m in $\Phi^{\supseteq}_{\vec{m}}$ has to be internalized (by universal quantification) to give $\Phi^{\supseteq}$ and the existentially quantified m inside $\Phi^{\subseteq}_{\vec{\sigma}}$ is made a parameter to yield $\Phi^{\subseteq}_{\vec{m},\sigma}$. Only the definition of

$\mu_I(\Phi)$ and $\mu_E(\Phi)$ will be shown here. For the other definitions and all the proofs see appendix A.
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

$$\Phi^{\supseteq} : \mathsf{SAT}_{\mu\alpha\rho} \to \mathsf{Pow}(\mathsf{SN}_{\rho[\alpha:=\mu\alpha\rho]})$$

via

$$\begin{array}{rcl} \Phi^{\supseteq}(\mathcal{M}) & := & \{t \in \mathsf{SN}_{\rho[\alpha:=\mu\alpha\rho]} | \forall m \in \forall\mathcal{P}\forall\mathcal{Q}.(\mathcal{P} \to \mathcal{Q}) \to \Phi(\mathcal{P}) \to \Phi(\mathcal{Q}) \\ & & \forall\sigma\forall\mathcal{N} \in \mathsf{SAT}_\sigma \forall s \in \mathcal{M} \to \mathcal{N}.m(\mu\alpha\rho)\sigma st \in \Phi_\sigma(\mathcal{N})\} \end{array}$$

and for $m \in \forall\mathcal{P}\forall\mathcal{Q}.(\mathcal{P} \to \mathcal{Q}) \to \Phi(\mathcal{P}) \to \Phi(\mathcal{Q})$ define $\Phi^{\subseteq}_{m} := (\Phi^{\subseteq}_{m,\sigma})_\sigma$ with

$$\Phi^{\subseteq}_{m,\sigma} : \mathsf{SAT}_\sigma \to \mathsf{SAT}_{\rho[\alpha:=\sigma]}$$

via

$$\begin{array}{rcl} \Phi^{\subseteq}_{m,\sigma}(\mathcal{N}) & := & \mathsf{cl}_{\rho[\alpha:=\sigma]}(\{r \in \Lambda_{\rho[\alpha:=\sigma]} | \exists\mathcal{M} \in \mathsf{SAT}_{\mu\alpha\rho} \exists s \in \mathcal{M} \to \mathcal{N} \exists t \in \Phi_{\mu\alpha\rho}(\mathcal{M}). \\ & & r = m(\mu\alpha\rho)\sigma st\}) \end{array}$$

$\Phi^{\supseteq}$ and $\Phi^{\subseteq}_{m,\sigma}$ are monotone.
Define for $\mathcal{M} \in \mathsf{SAT}_{\mu\alpha\rho}$

$$\begin{array}{rcl} M_I^{\supseteq}(\mathcal{M}) & := & \{r \in \Lambda_{\mu\alpha\rho} | \exists t \in \Phi^{\supseteq}(\mathcal{M}).r = C_{\mu\alpha\rho}t\} \\ M_E^{\subseteq}(\mathcal{M}) & := & \{r \in \Lambda_{\mu\alpha\rho} | \forall m \in \forall\mathcal{P}\forall\mathcal{Q}.(\mathcal{P} \to \mathcal{Q}) \to \Phi(\mathcal{P}) \to \Phi(\mathcal{Q}) \forall\sigma\forall\mathcal{N} \in \mathsf{SAT}_\sigma \\ & & \forall s \in \Phi_\sigma(\mathcal{N}) \to \mathcal{N}\, r\mathrm{E}_\mu\sigma ms \in \mathcal{N} \wedge \forall s \in \Phi^{\subseteq}_{m,\mu\alpha\rho\times\sigma}(\mathcal{M} \times \mathcal{N}) \to \mathcal{N}\, r\mathrm{E}^+_\mu\sigma ms \in \mathcal{N}\} \end{array}$$

and set

$$\mu_I(\Phi) := \text{ the least fixed point of } \Psi_I^{\supseteq} : \mathsf{SAT}_{\mu\alpha\rho} \to \mathsf{SAT}_{\mu\alpha\rho},$$

defined by

$$\Psi_I^{\supseteq}(\mathcal{M}) := \mathsf{cl}_{\mu\alpha\rho}(M_I^{\supseteq}(\mathcal{M}))$$

and

$$\mu_E(\Phi) := \text{ any fixed point of } \Psi_E^{\subseteq} : \mathsf{SAT}_{\mu\alpha\rho} \to \mathsf{SAT}_{\mu\alpha\rho},$$

defined by

$$\Psi_E^{\subseteq}(\mathcal{M}) := \mathsf{cl}_{\mu\alpha\rho}(M_E^{\subseteq}(\mathcal{M})).$$

Because Ψ_I and Ψ_E are monotone (they are even non-strictly positive) and $\mathsf{SAT}_{\mu\alpha\rho}$ is a complete lattice, such fixed points exists. Hence, $\mu_I(\Phi)$ and $\mu_E(\Phi)$ are $\mu\alpha\rho$-saturated sets.
In appendix A it will be shown that in fact the proof of $\mathsf{SN} = \Lambda$ given for EMIT goes through without changes to the content of it. Clearly, this reflects that β-reduction for μ-types only applies if the introduction and the elimination rule come together and then the monotonicity witness is used in the same way for EMIT and for IMIT.

9.5.4 Strong normalization of varIMIT

Extend the proof for IMIT to varIMIT by refining it in the same spirit as the proof for EMIT is refined to cover varEMIT. This time only the introduction-based approach directly generalizes. For the elimination-based definition another idea has to be adopted: We essentially use the general method of defining $\Psi_E^{\subseteq}$ as a monotone operator being pointwise smaller than Ψ_E presented in the introduction to chapter 6. Here only the definitions which show the construction of $\mu_I(\Phi)$ and $\mu_E(\Phi)$ are shown. All the details may be found in appendix A.

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

$$\Phi^{\supseteq} : \mathsf{SAT}_{\mu\alpha\rho} \to \mathsf{Pow}(\mathsf{SN}_{\rho[\alpha:=\mu\alpha\rho]})$$

via

$$\begin{array}{rcl} \Phi^{\supseteq}(\mathcal{M}) & := & \{t \in \mathsf{SN}_{\rho[\alpha:=\mu\alpha\rho]} | \forall\sigma\forall\mathcal{N} \in \mathsf{SAT}_\sigma \forall m \in \forall\mathcal{P}.(\mathcal{P} \to \mathcal{N}) \to \Phi(\mathcal{P}) \to \Phi_\sigma(\mathcal{N}) \\ & & \forall s \in \mathcal{M} \to \mathcal{N}.m(\mu\alpha\rho)st \in \Phi_\sigma(\mathcal{N})\} \end{array}$$

$\Phi^{\supseteq}$ is monotone (even non-strictly positive) and $\Phi^{\supseteq} \supseteq \Phi_{\mu\alpha\rho}$ (pointwise) because of the rules ($\forall$-E) and ($\to$-E) for saturated sets.
Define for $\mathcal{M} \in \mathsf{SAT}_{\mu\alpha\rho}$

$$\begin{array}{rcl} M_I^{\supseteq}(\mathcal{M}) & := & \{r \in \Lambda_{\mu\alpha\rho} | \exists t \in \Phi^{\supseteq}(\mathcal{M}).r = C_{\mu\alpha\rho}t\} \\ M_E^{\subseteq}(\mathcal{M}) & := & \{r \in \Lambda_{\mu\alpha\rho} | \forall\sigma\forall\mathcal{N} \in \mathsf{SAT}_\sigma \\ & & \forall m \in \forall\mathcal{P}.(\mathcal{P} \to \mathcal{N}) \to \Phi(\mathcal{P}) \to \Phi_\sigma(\mathcal{N}) \\ & & \forall s \in \Phi_\sigma(\mathcal{N}) \to \mathcal{N}\, r\mathrm{E}_\mu\sigma ms \in \mathcal{N} \wedge \\ & & \forall\mathcal{M}' \in \mathsf{SAT}_{\mu\alpha\rho}.\mathcal{M} \subseteq \mathcal{M}' \Rightarrow \\ & & \forall m \in \forall\mathcal{P}.(\mathcal{P} \to \mathcal{M}' \times \mathcal{N}) \to \Phi(\mathcal{P}) \to \Phi_{\mu\alpha\rho\times\sigma}(\mathcal{M}' \times \mathcal{N}) \\ & & \forall s \in \Phi_{\mu\alpha\rho\times\sigma}(\mathcal{M}' \times \mathcal{N}) \to \mathcal{N}\, r\mathrm{E}^+_\mu\sigma ms \in \mathcal{N}\} \end{array}$$

and set

$$\mu_I(\Phi) := \text{ the least fixed point of } \Psi_I^{\supseteq} : \mathsf{SAT}_{\mu\alpha\rho} \to \mathsf{SAT}_{\mu\alpha\rho},$$

defined by

$$\Psi_I^{\supseteq}(\mathcal{M}) := \mathsf{cl}_{\mu\alpha\rho}(M_I^{\supseteq}(\mathcal{M}))$$

and

$$\mu_E(\Phi) := \text{ any fixed point of } \Psi_E^{\subseteq} : \mathsf{SAT}_{\mu\alpha\rho} \to \mathsf{SAT}_{\mu\alpha\rho},$$

defined by

$$\Psi_E^{\subseteq}(\mathcal{M}) := \mathsf{cl}_{\mu\alpha\rho}(M_E^{\subseteq}(\mathcal{M})).$$

Because Ψ_I and Ψ_E are monotone (they are even non-strictly positive) and $\mathsf{SAT}_{\mu\alpha\rho}$ is a complete lattice, such fixed points exists. Hence, $\mu_I(\Phi)$ and $\mu_E(\Phi)$ are $\mu\alpha\rho$-saturated sets.
See appendix A for all the details leading to $\mathsf{SN} = \Lambda$ and discussion of the failure of the generalization of the elimination-based proof for IMIT.

9.6 Strong normalization of the other systems

In chapter 7 all the embeddings claimed in *the* main theorem in chapter 1 have been proved. Hence every system embeds into any of the systems in class III. Therefore due to Lemma 2.49 we only needed to prove strong normalization for one of those, e. g. for MeIT−it, and this has been done in this chapter twice: with the help of introduction-based computability predicates and with elimination-based such predicates.
The embeddings only show that every term is in sn. But it has been always easy to prove that $\mathsf{sn} \subseteq \mathsf{SN}$ and hence we also get the induction principle of SN and because $\mathsf{SN} \subseteq \mathsf{WN}$ also the induction principle of WN which allows to define the normalization function Ω.
Also Corollary 2.43 (concerning the Peirce formula) may be proved in all of the systems with the same proof (as was the case for Lemma 2.37 and 2.46 expressing consistency).

Although we now know strong normalization of PIT by indirect means it is interesting to think of a direct proof. It would work as indicated in the footnote 14 on p. 129: For the construction of the fixed points $\mu_I(\Phi)$ and $\mu_E(\Phi)$ one would require that every Φ_τ be monotone. This monotonicity has to be preserved by any of the constructions and hence one would be forced to take the least or greatest fixed point in the elimination-based approach. The essential lemmas which say that $\mu_I(\Phi)$ is a post-fixed-point of Ψ_E and that $\mu_E(\Phi)$ is a pre-fixed-point of Ψ_I can only be proved under the assumption that already $\mathsf{map}_{\lambda\alpha\rho} \in \forall\mathcal{P}\forall\mathcal{Q}.(\mathcal{P} \to \mathcal{Q}) \to \Phi(\mathcal{P}) \to \Phi(\mathcal{Q})$. The definition of computability predicates would be done simultaneously with the proof that the function $\mathcal{P} \mapsto \mathsf{SC}_\rho[\vec{\alpha}, \alpha := \vec{\mathcal{P}}, \mathcal{P}]$ is monotone for $\alpha \in \mathsf{Pos}(\rho)$ and antitone for $\alpha \in \mathsf{Neg}(\rho)$ which of course needs the preservation of monotonicity of the construction of $\mu_E(\Phi)$. The final soundness lemma (stating strong computability under substitution) cannot be proved by structural recursion because of the assumptions on $\mathsf{map}_{\lambda\alpha\rho}$ in the crucial lemmas. If $\mu_I(\Phi)$ is used, the proof goes by induction on the set ST of stratified terms of ISMIT (of which PIT is an instance) and in case $\mu_E(\Phi)$ is chosen, induction on ST of ESMIT works.
For NIPIT a direct proof would be a lot easier: Because there is no interleaving, $\mu_E(\Phi)$ need not be monotone in Φ. Hence, we may take any fixed point of Ψ_E as usual. The definition of computability predicates would be done simultaneously with the proof that the function $\mathcal{P} \mapsto \mathsf{SC}_\rho[\vec{\alpha}, \alpha := \vec{\mathcal{P}}, \mathcal{P}]$ is monotone for $\alpha \in \mathsf{NIPos}(\rho)$ and antitone for $\alpha \in \mathsf{NINeg}(\rho)$ and constant if $\alpha \notin \mathsf{FV}(\rho)$ (this already includes the coincidence lemma). Although the crucial lemmas have the same condition as in the proof for PIT, no reference to ST is needed because $\mathsf{map}_{\lambda\alpha\rho}$ does not use the term rules pertaining to μ-types: Simply show the soundness lemma first for terms without the rules (μ-I), (μ-E) and (μ-E$^+$) and then for all terms by reusing the proof clauses for the other term rules.
I decided to show the other direct proofs because they give a lot of insight into the techniques of proving strong normalization and into ways of dealing with monotonicity beyond positivity. Moreover, the extracted embeddings deserve the reader's attention.

Appendix A

Other proofs of strong normalization

In this appendix direct proofs of strong normalization are studied for the systems varEMIT, IMIT and varIMIT. Also the extracted embeddings are given. In the final section some remarks on the different methods of monotonization are made and it is sketched how the embeddings of monotone inductive types into MePIT and coMePIT could be drawn from normalization proofs.

A.1 Proof of strong normalization for varEMIT

This section contains the elimination-based proof for varEMIT which is only a minor variant to the proof for EMIT. The introduction-based proof could not directly be generalized to varEMIT. But an introduction-based proof following the general idea of strict positivization already used for MeIT–it is also presented.

Extend the definition of SN for system eF by the expected clauses:

(μ-I) If $C_{\mu\alpha\rho}mt$ is a term and $m \in \mathsf{SN}$ and $t \in \mathsf{SN}$, then $C_{\mu\alpha\rho}mt \in \mathsf{SN}$.

(β_μ) If $s\Big(m\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma s)t\Big)\vec{s} \in \mathsf{SN}$, $x^{\mu\alpha\rho} \notin \mathsf{FV}(s)$ and $(C_{\mu\alpha\rho}mt)\mathrm{E}_\mu\sigma s\vec{s}$ is a term, then $(C_{\mu\alpha\rho}mt)\mathrm{E}_\mu\sigma s\vec{s} \in \mathsf{SN}$.

(β_μ^+) If $s\Big(m(\mu\alpha\rho \times \sigma)\Big(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma s)x\rangle\Big)t\Big)\vec{s} \in \mathsf{SN}$, $x^{\mu\alpha\rho} \notin \mathsf{FV}(s)$ and the expression $(C_{\mu\alpha\rho}mt)\mathrm{E}_\mu^+\sigma s\vec{s}$ is a term, then $(C_{\mu\alpha\rho}mt)\mathrm{E}_\mu^+\sigma s\vec{s} \in \mathsf{SN}$.

Lemma A.1 If $m \in \mathsf{sn}$ and $t \in \mathsf{sn}$, then $C_{\mu\alpha\rho}mt \in \mathsf{sn}$.

Proof Induction on $m \in \mathsf{sn}$ and $t \in \mathsf{sn}$. □

Lemma A.2 If $s\Big(m\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma s)t\Big)\vec{s} \in \mathsf{sn}$, $x^{\mu\alpha\rho} \notin \mathsf{FV}(s)$ and $(C_{\mu\alpha\rho}mt)\mathrm{E}_\mu\sigma s\vec{s}$ is a term, then $(C_{\mu\alpha\rho}mt)\mathrm{E}_\mu\sigma s\vec{s} \in \mathsf{sn}$.

Proof Induction on $s\Big(m\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma s)t\Big)\vec{s} \in \mathsf{sn}^+$ by the usual analysis of the reducts of $(C_{\mu\alpha\rho}mt)\mathrm{E}_\mu\sigma s\vec{s}$. (Note that we need sn^+ because s occurs twice in the canonical reduct of $(C_{\mu\alpha\rho}mt)\mathrm{E}_\mu\sigma s\vec{s}$.) □

Lemma A.3 If $s\Big(m(\mu\alpha\rho \times \sigma)\Big(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma s)x\rangle\Big)t\Big)\vec{s} \in \mathsf{sn}$, $x^{\mu\alpha\rho} \notin \mathsf{FV}(s)$ and the expression $(C_{\mu\alpha\rho}mt)\mathrm{E}_\mu^+\sigma s\vec{s}$ is a term, then $(C_{\mu\alpha\rho}mt)\mathrm{E}_\mu^+\sigma s\vec{s} \in \mathsf{sn}$.

Proof Induction on $s\Big(m(\mu\alpha\rho \times \sigma)\Big(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma s)x\rangle\Big)t\Big)\vec{s} \in \mathsf{sn}^+$ by analysis of the reducts of $(C_{\mu\alpha\rho}mt)\mathrm{E}_\mu^+\sigma s\vec{s}$. (Note that we need sn^+ because s occurs twice in the canonical reduct of $(C_{\mu\alpha\rho}mt)\mathrm{E}_\mu^+\sigma s\vec{s}$.) □

Corollary A.4 $\mathsf{SN} \subseteq \mathsf{sn}$.

Proof The same as for Lemma 9.6. □

A.1.1 Saturated sets for varEMIT

Extend the definition for system eF of $\mathcal{M}$ being a τ-saturated set by the following clause:

(β_μ) If $s\Big(m\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma s)t\Big)\vec{s} \in \mathcal{M}$, $x^{\mu\alpha\rho} \notin \mathsf{FV}(s)$ and $(C_{\mu\alpha\rho}mt)\mathrm{E}_\mu\sigma s\vec{s}$ is a term, then $(C_{\mu\alpha\rho}mt)\mathrm{E}_\mu\sigma s\vec{s} \in \mathcal{M}$.

(β_μ^+) If $s\Big(m(\mu\alpha\rho \times \sigma)\Big(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma s)x\rangle\Big)t\Big)\vec{s} \in \mathcal{M}$, $x^{\mu\alpha\rho} \notin \mathsf{FV}(s)$ and the expression $(C_{\mu\alpha\rho}mt)\mathrm{E}_\mu^+\sigma s\vec{s}$ is a term, then $(C_{\mu\alpha\rho}mt)\mathrm{E}_\mu^+\sigma s\vec{s} \in \mathcal{M}$.

Extend the definition of the τ-saturated closure $\mathsf{cl}_\tau(M)$ accordingly by the clause

(β_μ) If $s\Big(m\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma s)t\Big)\vec{s} \in \mathsf{cl}_\tau(M)$, $x^{\mu\alpha\rho} \notin \mathsf{FV}(s)$ and $(C_{\mu\alpha\rho}mt)\mathrm{E}_\mu\sigma s\vec{s}$ is a term, then $(C_{\mu\alpha\rho}mt)\mathrm{E}_\mu\sigma s\vec{s} \in \mathsf{cl}_\tau(M)$.

(β_μ^+) If $s\Big(m(\mu\alpha\rho \times \sigma)\Big(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma s)x\rangle\Big)t\Big)\vec{s} \in \mathsf{cl}_\tau(M)$, $x^{\mu\alpha\rho} \notin \mathsf{FV}(s)$ and $(C_{\mu\alpha\rho}mt)\mathrm{E}_\mu^+\sigma s\vec{s}$ is a term, then $(C_{\mu\alpha\rho}mt)\mathrm{E}_\mu^+\sigma s\vec{s} \in \mathsf{cl}_\tau(M)$.

Again $\mathsf{cl}_\tau(M)$ is the smallest τ-saturated set containing $M \cap \mathsf{SN}_\tau$.

A.1.2 Calculating with saturated sets for varEMIT

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

$$
\begin{array}{rcl}
M_I(\mathcal{M}) & := & \{r \in \Lambda_{\mu\alpha\rho} | \exists m \in \forall\mathcal{Q}.(\mathcal{M} \to \mathcal{Q}) \to \Phi_{\mu\alpha\rho}(\mathcal{M}) \to \Phi(\mathcal{Q})\exists t \in \Phi_{\mu\alpha\rho}(\mathcal{M}).r = C_{\mu\alpha\rho}mt\} \\
M_E(\mathcal{M}) & := & \{r \in \Lambda_{\mu\alpha\rho} | \forall\sigma\forall\mathcal{N} \in \mathsf{SAT}_\sigma.\forall s \in \Phi_\sigma(\mathcal{N}) \to \mathcal{N}\, r\mathrm{E}_\mu\sigma s \in \mathcal{N} \wedge \\
 & & \qquad \forall s \in \Phi_{\mu\alpha\rho\times\sigma}(\mathcal{M} \times \mathcal{N}) \to \mathcal{N}\, r\mathrm{E}_\mu^+\sigma s \in \mathcal{N}\}
\end{array}
$$

and define $\Psi_I : \mathsf{SAT}_{\mu\alpha\rho} \to \mathsf{SAT}_{\mu\alpha\rho}$ by

$$\Psi_I(\mathcal{M}) := \mathsf{cl}_{\mu\alpha\rho}(M_I(\mathcal{M}))$$

and $\Psi_E : \mathsf{SAT}_{\mu\alpha\rho} \to \mathsf{SAT}_{\mu\alpha\rho}$ by

$$\Psi_E(\mathcal{M}) := \mathsf{cl}_{\mu\alpha\rho}(M_E(\mathcal{M})).$$

We again cannot ensure Ψ_I or Ψ_E to be monotone. In order to overcome this problem we define (with Pow denoting the power set)

$$\Phi^{\supseteq} : \mathsf{SAT}_{\mu\alpha\rho} \to \mathsf{Pow}(\mathsf{SN}_{\rho[\alpha:=\mu\alpha\rho]})$$

via

$$
\begin{array}{rcl}
\Phi^{\supseteq}(\mathcal{M}) & := & \{t \in \mathsf{SN}_{\rho[\alpha:=\mu\alpha\rho]} | \forall m \in \forall\mathcal{Q}.(\mathcal{M} \to \mathcal{Q}) \to \Phi_{\mu\alpha\rho}(\mathcal{M}) \to \Phi(\mathcal{Q}) \\
 & & \qquad \forall\sigma\forall\mathcal{N} \in \mathsf{SAT}_\sigma \forall s \in \mathcal{M} \to \mathcal{N}.m\sigma st \in \Phi_\sigma(\mathcal{N})\}
\end{array}
$$

and $\Phi^{\subseteq} := (\Phi^{\subseteq}_{\sigma})_\sigma$ with

$$\Phi^{\subseteq}_{\sigma} : \mathsf{SAT}_\sigma \to \mathsf{SAT}_{\rho[\alpha:=\sigma]}$$

via

$$\begin{array}{rcl}\Phi^{\subseteq}_{\sigma}(\mathcal{N}) & := & \mathsf{cl}_{\rho[\alpha:=\sigma]}(\{r \in \Lambda_{\rho[\alpha:=\sigma]} | \exists \mathcal{M} \in \mathsf{SAT}_{\mu\alpha\rho} \exists m \in \forall \mathcal{Q}.(\mathcal{M} \to \mathcal{Q}) \to \Phi_{\mu\alpha\rho}(\mathcal{M}) \to \Phi(\mathcal{Q}) \\ & & \exists s \in \mathcal{M} \to \mathcal{N} \exists t \in \Phi_{\mu\alpha\rho}(\mathcal{M}).r = m\sigma st\})\end{array}$$

$\Phi^{\supseteq}$ is presumably not monotone (in general). $\Phi^{\supseteq} \supseteq \Phi_{\mu\alpha\rho}$ (pointwise) because of the rules ($\forall$-E) and ($\to$-E) for saturated sets. $\Phi^{\subseteq}_{\sigma}$ is monotone (even strictly positive) and $\Phi^{\subseteq}_{\sigma} \subseteq \Phi_\sigma$ (pointwise) also because of the rules ($\forall$-E) and ($\to$-E) for saturated sets. (Note that already the set which is the argument to $\mathsf{cl}_{\rho[\alpha:=\sigma]}$ is a subset of $\Phi_\sigma(\mathcal{M})$ and therefore also $\Phi^{\subseteq}_{\sigma}(\mathcal{M})$ because $\Phi_\sigma(\mathcal{M})$ is saturated.)
Define for $\mathcal{M} \in \mathsf{SAT}_{\mu\alpha\rho}$

$$\begin{array}{rcl}M^{\supseteq}_{I}(\mathcal{M}) & := & \{r \in \Lambda_{\mu\alpha\rho} | \exists m \in \forall \mathcal{Q}.(\mathcal{M} \to \mathcal{Q}) \to \Phi_{\mu\alpha\rho}(\mathcal{M}) \to \Phi(\mathcal{Q}) \exists t \in \Phi^{\supseteq}(\mathcal{M}).r = C_{\mu\alpha\rho}mt\} \\ M^{\subseteq}_{E}(\mathcal{M}) & := & \{r \in \Lambda_{\mu\alpha\rho} | \forall\sigma\forall\mathcal{N} \in \mathsf{SAT}_\sigma.\forall s \in \Phi_\sigma(\mathcal{N}) \to \mathcal{N}\, r\mathrm{E}_\mu \sigma s \in \mathcal{N} \wedge \\ & & \forall s \in \Phi^{\subseteq}_{\mu\alpha\rho\times\sigma}(\mathcal{M} \times \mathcal{N}) \to \mathcal{N}\, r\mathrm{E}^{+}_{\mu}\sigma s \in \mathcal{N}\}\end{array}$$

and define

$$\Psi^{\supseteq}_{I} : \mathsf{SAT}_{\mu\alpha\rho} \to \mathsf{SAT}_{\mu\alpha\rho},$$

by

$$\Psi^{\supseteq}_{I}(\mathcal{M}) := \mathsf{cl}_{\mu\alpha\rho}(M^{\supseteq}_{I}(\mathcal{M}))$$

and set

$$\mu_E(\Phi) := \text{ any fixed point of } \Psi^{\subseteq}_{E} : \mathsf{SAT}_{\mu\alpha\rho} \to \mathsf{SAT}_{\mu\alpha\rho},$$

defined by

$$\Psi^{\subseteq}_{E}(\mathcal{M}) := \mathsf{cl}_{\mu\alpha\rho}(M^{\subseteq}_{E}(\mathcal{M})).$$

Because Ψ_E is monotone (it is even non-strictly positive) and $\mathsf{SAT}_{\mu\alpha\rho}$ is a complete lattice, the fixed point exists. Hence, $\mu_E(\Phi)$ is a $\mu\alpha\rho$-saturated set. $\Psi^{\supseteq}_{I}$ cannot be monotone in general (not only the problem with $\Phi^{\supseteq}$ propagates but also $M^{\supseteq}_{I}(\mathcal{M})$'s definition has several critical occurrences of $\mathcal{M}$). The definitions of $\Phi^{\supseteq}$, $M^{\supseteq}_{I}(\mathcal{M})$ and $\Psi^{\supseteq}_{I}$ are thus useless. $\Phi^{\supseteq}$ will not be used any further and $M^{\supseteq}_{I}(\mathcal{M})$ and $\Psi^{\supseteq}_{I}$ are redefined in the following way:

$$\begin{array}{rcl}M^{\supseteq}_{I}(\mathcal{M}) & := & \{r \in \Lambda_{\mu\alpha\rho} | \exists \mathcal{M}' \in \mathsf{SAT}_{\mu\alpha\rho}.\mathcal{M}' \subseteq \mathcal{M} \wedge \\ & & \exists m \in \forall \mathcal{Q}.(\mathcal{M}' \to \mathcal{Q}) \to \Phi_{\mu\alpha\rho}(\mathcal{M}') \to \Phi(\mathcal{Q}) \exists t \in \Phi_{\mu\alpha\rho}(\mathcal{M}').r = C_{\mu\alpha\rho}mt\} \\ \Psi^{\supseteq}_{I}(\mathcal{M}) & := & \mathsf{cl}_{\mu\alpha\rho}(M^{\supseteq}_{I}(\mathcal{M}))\end{array}$$

$\Psi^{\supseteq}_{I}$ is monotone (even strictly positive) and trivially, $\Psi^{\supseteq}_{I} \supseteq \Psi_I$ (pointwise). We may now form

$$\mu_I(\Phi) := \text{ the least fixed point of } \Psi^{\supseteq}_{I},$$

which therefore is also a pre-fixed-point of Ψ_I.
An analysis of the definition of $\mu_I(\Phi)$ gives the following encoding (for "new" β and γ):

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

An embedding of $\mathsf{varEMIT}$ into $\mathsf{NISPIT{+}ex}$ may now be read off the proofs that $\mu_I(\Phi)$ is a pre-fixed-point of Ψ_I and that $\mu_I(\Phi)$ is a post-fixed-point of Ψ_E[1] which are given below. It is possible to show that this indeed gives an embedding.

[1] In fact we prove that $\mu_I(\Phi)$ is a post-fixed-point of $\Psi^{\subseteq}_{E}$ but the proof of the weaker statement has exactly the same combinatorial content.

The elimination-based approach presumably gives a quite complicated embedding into non-interleaving positive fixed-point types based on
$(\mu\alpha\rho)' := \hat{\mu}\alpha\forall\beta.((\rho'[\alpha := \beta] \to \beta) \to \beta)\times$
$\quad(((\exists\gamma.(\forall\delta.(\gamma \to \delta) \to \rho'[\alpha := \gamma] \to \rho'[\alpha := \delta]) \times (\gamma \to \alpha \times \beta) \times \rho'[\alpha := \gamma]) \to \beta) \to \beta)$
with "new" β, γ and δ[2].

Lemma A.5 $M_E(\mathcal{M}) \cap \mathsf{SN} \in \mathsf{SAT}$ and $M^{\subseteq}_{E}(\mathcal{M}) \cap \mathsf{SN} \in \mathsf{SAT}$ (which implies that $\Psi_E(\mathcal{M}) = M_E(\mathcal{M}) \cap \mathsf{SN}$ and $\Psi^{\subseteq}_{E}(\mathcal{M}) = M^{\subseteq}_{E}(\mathcal{M}) \cap \mathsf{SN}$).

Proof Straightforward. □

Lemma A.6 $M_I(\mathcal{M}) \subseteq \mathsf{SN}$ and $M^{\supseteq}_{I}(\mathcal{M}) \subseteq \mathsf{SN}$ (which implies that $\Psi_I(\mathcal{M}) \supseteq M_I(\mathcal{M})$ and $\Psi^{\supseteq}_{I}(\mathcal{M}) \supseteq M^{\supseteq}_{I}(\mathcal{M})$). □

Lemma A.7 Let $\mathcal{M}$ be a $\mu\alpha\rho$-saturated set $\mathcal{M}$.
$\mathcal{M}$ is a pre-fixed-point of Ψ_I iff for all $m \in \forall\mathcal{Q}.(\mathcal{M} \to \mathcal{Q}) \to \Phi_{\mu\alpha\rho}(\mathcal{M}) \to \Phi(\mathcal{Q})$ and for all $t \in \Phi_{\mu\alpha\rho}(\mathcal{M})$, we have that $C_{\mu\alpha\rho}mt \in \mathcal{M}$.
$\mathcal{M}$ is a post-fixed-point of Ψ_E iff for all $r \in \mathcal{M}$, types σ, $\mathcal{N} \in \mathsf{SAT}_\sigma$ and $s \in \Phi_\sigma(\mathcal{N}) \to \mathcal{N}$, we have $r\mathrm{E}_\mu\sigma s \in \mathcal{N}$ and for all $s \in \Phi_{\mu\alpha\rho\times\sigma}(\mathcal{M} \times \mathcal{N}) \to \mathcal{N}$, $r\mathrm{E}^{+}_{\mu}\sigma s \in \mathcal{N}$.

Proof Use the two preceding lemmas. □

Lemma A.8 Every post-fixed-point of $\Psi^{\subseteq}_{E}$ is also a post-fixed-point of Ψ_E.

Proof Let $\mathcal{M}$ be a post-fixed-point of $\Psi^{\subseteq}_{E}$. Hence

$$\mathcal{M} \subseteq \Psi^{\subseteq}_{E}(\mathcal{M}) \subseteq \Psi_E(\mathcal{M}),$$

which clearly follows from $\Phi^{\subseteq} \subseteq \Phi$ (typewise) because $M_E(\mathcal{M})$ and $M^{\subseteq}_{E}(\mathcal{M})$ have the same definition with exception of the occurrence of $\Phi_{\mu\alpha\rho\times\sigma}$ and $\Phi^{\subseteq}_{\mu\alpha\rho\times\sigma}$, respectively, which is at a non-strictly positive position. □

Lemma A.9 $\mu_E(\Phi)$ is a pre-fixed-point of $\Psi^{\supseteq}_{I}$.

Proof We have to show that

$$\Psi^{\supseteq}_{I}(\mu_E(\Phi)) \subseteq \mu_E(\Phi).$$

Because $\mu_E(\Phi)$ is a pre-fixed-point of $\Psi^{\subseteq}_{E}$, it suffices to show

$$\Psi^{\supseteq}_{I}(\mu_E(\Phi)) \subseteq \Psi^{\subseteq}_{E}(\mu_E(\Phi)).$$

Because $\mathsf{cl}_{\mu\alpha\rho}$ is monotone, it suffices to show

$$M^{\supseteq}_{I}(\mu_E(\Phi)) \subseteq M^{\subseteq}_{E}(\mu_E(\Phi)).$$

Let $\mathcal{M} \in \mathsf{SAT}_{\mu\alpha\rho}$ with $\mathcal{M} \subseteq \mu_E(\Phi)$. Let $m \in \forall\mathcal{Q}.(\mathcal{M} \to \mathcal{Q}) \to \Phi_{\mu\alpha\rho}(\mathcal{M}) \to \Phi(\mathcal{Q})$ and $t \in \Phi_{\mu\alpha\rho}(\mathcal{M})$. We have to show

$$C_{\mu\alpha\rho}mt \in M^{\subseteq}_{E}(\mu_E(\Phi)).$$

[2] If we only had iteration then $(\mu\alpha\rho)' := \forall\beta.((\rho'[\alpha := \beta] \to \beta) \to \beta)$ would be sufficient which is nothing but the composition of the embeddings of $\mathsf{varEMIT{-}rec}$ into IT and IT into eF.

Let σ be a type, $\mathcal{N} \in \mathsf{SAT}_\sigma$ and $s \in \Phi_\sigma(\mathcal{N}) \to \mathcal{N}$. We have to show that

$$C_{\mu\alpha\rho} mt \mathrm{E}_\mu \sigma s \in \mathcal{N}.$$

Because $\mathcal{N}$ is saturated it suffices to show

$$s\Big(m\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma s)t\Big) \in \mathcal{N}.$$

Because of ($\to$-E) for saturated sets (in the sequel the named rules will always be those for saturated sets) it suffices to show

$$m\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma s)t \in \Phi_\sigma(\mathcal{N}).$$

By ($\forall$-E) and ($\to$-E) it suffices to show

$$\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma s \in \mathcal{M} \to \mathcal{N}.$$

We use ($\to$-I). Let $r \in \mathcal{M}$. We have to show

$$r\mathrm{E}_\mu\sigma s \in \mathcal{N}.$$

This follows from $r \in \mathcal{M} \subseteq \mu_E(\Phi) \subseteq \Psi_E^{\subseteq}(\mu_E(\Phi)) \subseteq M_E^{\subseteq}(\mu_E(\Phi))$.
We come to the case for which $\Psi_E^{\subseteq}$ was designed, namely $s \in \Phi^{\subseteq}_{\mu\alpha\rho\times\sigma}(\mu_E(\Phi) \times \mathcal{N}) \to \mathcal{N}$. We have to show

$$C_{\mu\alpha\rho} mt \mathrm{E}^+_\mu \sigma s \in \mathcal{N}.$$

By saturatedness of $\mathcal{N}$ it suffices to show

$$s\Big(m(\mu\alpha\rho \times \sigma)\Big(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma s)x\rangle\Big)t\Big) \in \mathcal{N}.$$

Because of ($\to$-E) it suffices to show

$$m(\mu\alpha\rho \times \sigma)\Big(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma s)x\rangle\Big)t \in \Phi^{\subseteq}_{\mu\alpha\rho\times\sigma}(\mu_E(\Phi) \times \mathcal{N}).$$

According to the definition of $\Phi^{\subseteq}$ (the closure in the definition of $\Phi^{\subseteq}$ only makes the set larger) we are done if we show

$$\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma s)x\rangle \in \mathcal{M} \to \mu_E(\Phi) \times \mathcal{N}.$$

We use ($\to$-I). Let $r \in \mathcal{M}$. We have to show

$$\langle r, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma s)r\rangle \in \mu_E(\Phi) \times \mathcal{N}.$$

By ($\times$-I) and because $\mathcal{M} \subseteq \mu_E(\Phi)$ it suffices to show

$$(\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma s)r \in \mathcal{N}.$$

Because $\mathcal{N}$ is saturated, we only need to show

$$r\mathrm{E}^+_\mu\sigma s \in \mathcal{N}.$$

This follows again from $r \in \mathcal{M} \subseteq \mu_E(\Phi) \subseteq \Psi_E^{\subseteq}(\mu_E(\Phi)) \subseteq M_E^{\subseteq}(\mu_E(\Phi))$. □

Note that we used from $\mu_E(\Phi)$ the properties of being a pre-fixed-point and a post-fixed-point of $\Psi_E^{\subseteq}$. Note also that for the purposes of the normalization proof it would have been enough to show that $\mu_E(\Phi)$ is a pre-fixed-point of Ψ_I instead of $\Psi_I^{\supseteq}$, but from the above result we even get the following

Corollary A.10 $\mu_I(\Phi) \subseteq \mu_E(\Phi)$.

Proof $\mu_I(\Phi)$ is the least pre-fixed-point of $\Psi_I^{\supseteq}$. □

Lemma A.11 $\mu_I(\Phi)$ is a post-fixed-point of $\Psi_E^{\subseteq}$.

Proof Set $\mathcal{M} := \mu_I(\Phi)$. We prove $\mathcal{M} \subseteq \Psi_E^{\subseteq}(\mathcal{M})$ by extended induction (as defined in the introduction to chapter 4) on $\mathcal{M}$. Let $\mathcal{M}' := \Psi_E^{\subseteq}(\mathcal{M}) \cap \mathcal{M}$. We have to show

$$\Psi_I^{\supseteq}(\mathcal{M}') \subseteq \Psi_E^{\subseteq}(\mathcal{M}).$$

Because $\mathsf{cl}_{\mu\alpha\rho}$ is monotone, it suffices to show

$$M_I^{\supseteq}(\mathcal{M}') \subseteq M_E^{\subseteq}(\mathcal{M}).$$

Let $r \in M_I^{\supseteq}(\mathcal{M}')$, i.e., $r = C_{\mu\alpha\rho}mt$ with $m \in \forall\mathcal{Q}.(\mathcal{M}'' \to \mathcal{Q}) \to \Phi(\mathcal{M}'') \to \Phi(\mathcal{Q})$ and $t \in \Phi_{\mu\alpha\rho}(\mathcal{M}'')$ with $\mathcal{M}'' \in \mathsf{SAT}_{\mu\alpha\rho}$ and $\mathcal{M}'' \subseteq \mathcal{M}'$. We have to show that

$$C_{\mu\alpha\rho}mt \in M_E^{\subseteq}(\mathcal{M}).$$

Let σ be a type, $\mathcal{N} \in \mathsf{SAT}_\sigma$ and $s \in \Phi_\sigma(\mathcal{N}) \to \mathcal{N}$. We have to show that

$$C_{\mu\alpha\rho} mt \mathrm{E}_\mu \sigma s \in \mathcal{N}.$$

Because $\mathcal{N}$ is saturated it suffices to show

$$s\Big(m\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma s)t\Big) \in \mathcal{N}.$$

Because of ($\to$-E) for saturated sets it suffices to show

$$m\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma s)t \in \Phi_\sigma(\mathcal{N}).$$

By ($\forall$-E) and ($\to$-E) it suffices to show

$$\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma s \in \mathcal{M}'' \to \mathcal{N}.$$

We use ($\to$-I). Let $r' \in \mathcal{M}''$. We have to show

$$r'\mathrm{E}_\mu\sigma s \in \mathcal{N}.$$

This follows from $\mathcal{M}'' \subseteq \mathcal{M}' \subseteq \Psi_E^{\subseteq}(\mathcal{M}) \subseteq M_E^{\subseteq}(\mathcal{M})$.
We come to the case which needs the extended form of induction. Let $s \in \Phi^{\subseteq}_{\mu\alpha\rho\times\sigma}(\mathcal{M}\times\mathcal{N}) \to \mathcal{N}$. We have to show

$$C_{\mu\alpha\rho} mt \mathrm{E}^+_\mu \sigma s \in \mathcal{N}.$$

By saturatedness of $\mathcal{N}$ it suffices to show

$$s\Big(m(\mu\alpha\rho \times \sigma)\Big(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma s)x\rangle\Big)t\Big) \in \mathcal{N}.$$

Because of (→-E) it suffices to show

$$m(\mu\alpha\rho\times\sigma)\Big(\lambda x^{\mu\alpha\rho}.\langle x,(\lambda x^{\mu\alpha\rho}.x\mathrm{E}^{+}_{\mu}\sigma s)x\rangle\Big)t\in\Phi^{\subseteq}_{\mu\alpha\rho\times\sigma}(\mathcal{M}\times\mathcal{N}).$$

By definition of $\Phi^{\subseteq}$ (the closure in the definition of $\Phi^{\subseteq}$ only makes the set larger) it suffices to show

$$\lambda x^{\mu\alpha\rho}.\langle x,(\lambda x^{\mu\alpha\rho}.x\mathrm{E}^{+}_{\mu}\sigma s)x\rangle\in\mathcal{M}''\to\mathcal{M}\times\mathcal{N}.$$

We use (→-I). Let $r'\in\mathcal{M}''$. We have to show

$$\langle r',(\lambda x^{\mu\alpha\rho}.x\mathrm{E}^{+}_{\mu}\sigma s)r'\rangle\in\mathcal{M}\times\mathcal{N}.$$

By (×-I) it suffices to show

$$r'\in\mathcal{M}\text{ and }(\lambda x^{\mu\alpha\rho}.x\mathrm{E}^{+}_{\mu}\sigma s)r'\in\mathcal{N}.$$

Because $\mathcal{N}$ is saturated and $\mathcal{M}''\subseteq\mathcal{M}'\subseteq\mathcal{M}$ (this argument needs the extension of induction), we only need to show

$$r'\mathrm{E}^{+}_{\mu}\sigma s\in\mathcal{N}.$$

This follows again from $\mathcal{M}''\subseteq\mathcal{M}'\subseteq\Psi^{\subseteq}_{E}(\mathcal{M})\subseteq M^{\subseteq}_{E}(\mathcal{M})$. □

Note that we only used that $\mu_I(\Phi)$ is a least pre-fixed-point. Note also that for the purposes of the normalization proof it would have been enough to show that $\mu_I(\Phi)$ is a post-fixed-point of Ψ_E instead of $\Psi^{\subseteq}_{E}$[3].
Putting everything together we get the following rules for reasoning with $\mu_I(\Phi)$ and $\mu_E(\Phi)$ where we simply write $\mu(\Phi)$ for both of them:

(SAT) $\mu(\Phi)\in\mathsf{SAT}_{\mu\alpha\rho}$.

(μ-I) If $m\in\forall\mathcal{Q}.(\mu(\Phi)\to\mathcal{Q})\to\Phi(\mu(\Phi))\to\Phi(\mathcal{Q})$ and $t\in\Phi_{\mu\alpha\rho}(\mu(\Phi))$, then $C_{\mu\alpha\rho}mt\in\mu(\Phi)$.

(μ-E) If $r\in\mu(\Phi)$, σ type, $\mathcal{N}\in\mathsf{SAT}_\sigma$ and $s\in\Phi_\sigma(\mathcal{N})\to\mathcal{N}$, then $r\mathrm{E}_\mu\sigma s\in\mathcal{N}$.

(μ-E$^+$) If $r\in\mu(\Phi)$, σ type, $\mathcal{N}\in\mathsf{SAT}_\sigma$ and $s\in\Phi_{\mu\alpha\rho\times\sigma}(\mu(\Phi)\times\mathcal{N})\to\mathcal{N}$, then $r\mathrm{E}^{+}_{\mu}\sigma s\in\mathcal{N}$.

A.1.3 Computability predicates for varEMIT

Extend (as for MeIT–it and coMeIT–it) the definition of $\mathsf{SC}_\rho[\vec\alpha:=\vec{\mathcal{P}}]$ by the following clause:

(μ) $\mathsf{SC}_{\mu\alpha\rho}[\vec\alpha:=\vec{\mathcal{P}}]:=\mu(\Phi)$ with $\Phi=(\Phi_\sigma)_\sigma$, where $\Phi_\sigma:\mathsf{SAT}_\sigma\to\mathsf{SAT}_{\rho[\vec\alpha,\alpha:=\vec\sigma,\sigma]}$ is defined by $\Phi_\sigma(\mathcal{P}):=\mathsf{SC}_\rho[\vec\alpha,\alpha:=\vec{\mathcal{P}},\mathcal{P}]$ (with the same assumptions as in the definition of strong computability for universal types in order to guarantee type-correctness).

The definition will again be written more intuitively as follows:

$$\mathsf{SC}_{\mu\alpha\rho}[\vec\alpha:=\vec{\mathcal{P}}]:=\mu\mathcal{P}.\mathsf{SC}_\rho[\vec\alpha,\alpha:=\vec{\mathcal{P}},\mathcal{P}].$$

It is obvious that coincidence and substitution lemma (Lemma 9.13 and Lemma 9.14) extend to varEMIT.
Now we extend Lemma 9.16 to varEMIT:

[3] If the greatest fixed point is taken for $\mu_E(\Phi)$, then also the preceding lemma implies that $\mu_I(\Phi)\subseteq\mu_E(\Phi)$.

Lemma A.12 Let r^ρ be a term, $\vec{x}^{\vec\rho}$ a list of variables and $\vec s$ an equally long list of terms such that $\vec s\in\mathsf{SC}_{\vec\rho}[\vec\alpha:=\vec{\mathcal{P}}^{\vec\sigma}]$ for some $\vec{\mathcal{P}}\subseteq\mathsf{SAT}$. Assume that

$$\forall x\forall\pi\forall\pi'.(x^\pi,x^{\pi'}\in\mathsf{FV}(r)\cup\vec{x}^{\vec\rho})\wedge(\pi[\vec\alpha:=\vec\sigma]=\pi'[\vec\alpha:=\vec\sigma])\Rightarrow\pi=\pi'.$$

Then $r[\vec\alpha:=\vec\sigma][\vec{x}^{\vec\rho[\vec\alpha:=\vec\sigma]}:=\vec s]\in\mathsf{SC}_\rho[\vec\alpha:=\vec{\mathcal{P}}]$.

Proof Induction on r. Again abbreviate $r[\vec\alpha:=\vec\sigma][\vec{x}^{\vec\rho[\vec\alpha:=\vec\sigma]}:=\vec s]$ by r' (for any term r). The assumption in the statement will again be referenced to as "injectivity". We only have to consider the rules (μ-I), (μ-E) and (μ-E$^+$).
(μ-I) Case $C_{\mu\alpha\rho}mt$. $(C_{\mu\alpha\rho}mt)'=C_{\mu\alpha\rho[\vec\alpha:=\vec\sigma]}m't'$. We may assume that $\alpha\notin\vec\alpha$. We have to show $C_{\mu\alpha\rho[\vec\alpha:=\vec\sigma]}m't'\in\mathsf{SC}_{\mu\alpha\rho}[\vec\alpha:=\vec{\mathcal{P}}]$. By the rule ($\mu$-I) for saturated sets it suffices to show

$$m'\in\forall\mathcal{Q}.(\mathsf{SC}_{\mu\alpha\rho}[\vec\alpha:=\vec{\mathcal{P}}]\to\mathcal{Q})\to\mathsf{SC}_\rho[\vec\alpha,\alpha:=\vec{\mathcal{P}},\mathsf{SC}_{\mu\alpha\rho}[\vec\alpha:=\vec{\mathcal{P}}]]\to\mathsf{SC}_\rho[\vec\alpha,\alpha:=\vec{\mathcal{P}},\mathcal{Q}]\text{ and}$$

$$t'\in\mathsf{SC}_\rho[\vec\alpha,\alpha:=\vec{\mathcal{P}},\mathsf{SC}_{\mu\alpha\rho}[\vec\alpha:=\vec{\mathcal{P}}]].$$

Because $\mathsf{FV}(C_{\mu\alpha\rho}mt)=\mathsf{FV}(m)\cup\mathsf{FV}(t)$, injectivity holds trivially also for m and t. By induction hypothesis $m'\in\mathsf{SC}_{\forall\alpha.(\mu\alpha\rho\to\alpha)\to\rho[\alpha:=\mu\alpha\rho]\to\rho}[\vec\alpha:=\vec{\mathcal{P}}]$ and $t'\in\mathsf{SC}_{\rho[\alpha:=\mu\alpha\rho]}[\vec\alpha:=\vec{\mathcal{P}}]$. By two applications of the coincidence lemma (which for system F is Lemma 9.13) we see that

$$\begin{aligned}&\mathsf{SC}_{\forall\alpha.(\mu\alpha\rho\to\alpha)\to\rho[\alpha:=\mu\alpha\rho]\to\rho}[\vec\alpha:=\vec{\mathcal{P}}]=\\&\qquad\forall\mathcal{Q}.(\mathsf{SC}_{\mu\alpha\rho}[\vec\alpha:=\vec{\mathcal{P}}]\to\mathcal{Q})\to\mathsf{SC}_{\rho[\alpha:=\mu\alpha\rho]}[\vec\alpha:=\vec{\mathcal{P}}]\to\mathsf{SC}_\rho[\vec\alpha,\alpha:=\vec{\mathcal{P}},\mathcal{Q}].\end{aligned}$$

By the substitution lemma (which for system F is Lemma 9.14) we have

$$\mathsf{SC}_{\rho[\alpha:=\mu\alpha\rho]}[\vec\alpha:=\vec{\mathcal{P}}]=\mathsf{SC}_\rho[\vec\alpha,\alpha:=\vec{\mathcal{P}},\mathsf{SC}_{\mu\alpha\rho}[\vec\alpha:=\vec{\mathcal{P}}]].$$

(μ-E) and (μ-E$^+$): Trivially the same as for EMIT. □

As for EMIT we get as corollaries that $\mathsf{SN}=\Lambda$ and that every term is strongly normalizing.

A.2 Proof of strong normalization for IMIT

This section contains the proof for IMIT which is only a minor variant to the proof for EMIT. Extend the definition of SN for system eF by the expected clauses:

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

(β_μ) If $s\Big(m(\mu\alpha\rho)\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma ms)t\Big)\vec s\in\mathsf{SN}$, $x^{\mu\alpha\rho}\notin\mathsf{FV}(s)$ and $(C_{\mu\alpha\rho}t)\mathrm{E}_\mu\sigma ms\vec s$ is a term, then $(C_{\mu\alpha\rho}t)\mathrm{E}_\mu\sigma ms\vec s\in\mathsf{SN}$.

(β^+_μ) If $s\Big(m(\mu\alpha\rho)(\mu\alpha\rho\times\sigma)\Big(\lambda x^{\mu\alpha\rho}.\langle x,(\lambda x^{\mu\alpha\rho}.x\mathrm{E}^{+}_{\mu}\sigma ms)x\rangle\Big)t\Big)\vec s\in\mathsf{SN}$, $x^{\mu\alpha\rho}\notin\mathsf{FV}(s)$ and $(C_{\mu\alpha\rho}t)\mathrm{E}^{+}_{\mu}\sigma ms\vec s$ is a term, then $(C_{\mu\alpha\rho}t)\mathrm{E}^{+}_{\mu}\sigma ms\vec s\in\mathsf{SN}$.

Lemma A.13 If $t\in\mathsf{sn}$, then $C_{\mu\alpha\rho}t\in\mathsf{sn}$.

Proof Induction on $t\in\mathsf{sn}$. □

Lemma A.14 If $s\Big(m(\mu\alpha\rho)\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma ms)t\Big)\vec s\in\mathsf{sn}$, $x^{\mu\alpha\rho}\notin\mathsf{FV}(s)$ and $(C_{\mu\alpha\rho}t)\mathrm{E}_\mu\sigma ms\vec s$ is a term, then $(C_{\mu\alpha\rho}t)\mathrm{E}_\mu\sigma ms\vec s\in\mathsf{sn}$.

Proof Induction on $s\Big(m(\mu\alpha\rho)\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma ms)t\Big)\vec{s} \in \mathsf{sn}^+$ by the usual analysis of the reducts of $(C_{\mu\alpha\rho}t)\mathrm{E}_\mu\sigma ms\vec{s}$. (Note that we need sn^+ because s occurs twice in the canonical reduct of $(C_{\mu\alpha\rho}t)\mathrm{E}_\mu\sigma ms\vec{s}$.) □

Lemma A.15 If $s\Big(m(\mu\alpha\rho)(\mu\alpha\rho\times\sigma)\Big(\lambda x^{\mu\alpha\rho}.\langle x,(\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma ms)x\rangle\Big)t\Big)\vec{s} \in \mathsf{sn}$, $x^{\mu\alpha\rho} \notin \mathsf{FV}(s)$ and $(C_{\mu\alpha\rho}t)\mathrm{E}^+_\mu\sigma ms\vec{s}$ is a term, then $(C_{\mu\alpha\rho}t)\mathrm{E}^+_\mu\sigma ms\vec{s} \in \mathsf{sn}$.

Proof Induction on $s\Big(m(\mu\alpha\rho)(\mu\alpha\rho\times\sigma)\Big(\lambda x^{\mu\alpha\rho}.\langle x,(\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma ms)x\rangle\Big)t\Big)\vec{s} \in \mathsf{sn}^+$ by analysis of the reducts of $(C_{\mu\alpha\rho}t)\mathrm{E}^+_\mu\sigma ms\vec{s}$. (Note that we need sn^+ because s occurs twice in the canonical reduct of $(C_{\mu\alpha\rho}t)\mathrm{E}^+_\mu\sigma ms\vec{s}$.) □

Corollary A.16 $\mathsf{SN} \subseteq \mathsf{sn}$.

Proof The same as for Lemma 9.6. □

A.2.1 Saturated sets for IMIT

Extend the definition for system eF of $\mathcal{M}$ being a τ-saturated set by the following clause:

(β_μ) If $s\Big(m(\mu\alpha\rho)\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma ms)t\Big)\vec{s} \in \mathcal{M}$, $x^{\mu\alpha\rho} \notin \mathsf{FV}(s)$ and $(C_{\mu\alpha\rho}t)\mathrm{E}_\mu\sigma ms\vec{s}$ is a term, then $(C_{\mu\alpha\rho}t)\mathrm{E}_\mu\sigma ms\vec{s} \in \mathcal{M}$.

(β^+_μ) If $s\Big(m(\mu\alpha\rho)(\mu\alpha\rho\times\sigma)\Big(\lambda x^{\mu\alpha\rho}.\langle x,(\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma ms)x\rangle\Big)t\Big)\vec{s} \in \mathcal{M}$, $x^{\mu\alpha\rho} \notin \mathsf{FV}(s)$ and $(C_{\mu\alpha\rho}t)\mathrm{E}^+_\mu\sigma ms\vec{s}$ is a term, then $(C_{\mu\alpha\rho}t)\mathrm{E}^+_\mu\sigma ms\vec{s} \in \mathcal{M}$.

Extend the definition of the τ-saturated closure $\mathsf{cl}_\tau(M)$ accordingly by the clause

(β_μ) If $s\Big(m(\mu\alpha\rho)\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma ms)t\Big)\vec{s} \in \mathsf{cl}_\tau(M)$, $x^{\mu\alpha\rho} \notin \mathsf{FV}(s)$ and $(C_{\mu\alpha\rho}t)\mathrm{E}_\mu\sigma ms\vec{s}$ is a term, then $(C_{\mu\alpha\rho}t)\mathrm{E}_\mu\sigma ms\vec{s} \in \mathsf{cl}_\tau(M)$.

(β^+_μ) If $s\Big(m(\mu\alpha\rho)(\mu\alpha\rho\times\sigma)\Big(\lambda x^{\mu\alpha\rho}.\langle x,(\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma ms)x\rangle\Big)t\Big)\vec{s} \in \mathsf{cl}_\tau(M)$, $x^{\mu\alpha\rho} \notin \mathsf{FV}(s)$ and $(C_{\mu\alpha\rho}t)\mathrm{E}^+_\mu\sigma ms\vec{s}$ is a term, then $(C_{\mu\alpha\rho}t)\mathrm{E}^+_\mu\sigma ms\vec{s} \in \mathsf{cl}_\tau(M)$.

Again $\mathsf{cl}_\tau(M)$ is the smallest τ-saturated set containing $M \cap \mathsf{SN}_\tau$.

A.2.2 Calculating with saturated sets for IMIT

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

$$\begin{array}{rcl} M_I(\mathcal{M}) & := & \{r \in \Lambda_{\mu\alpha\rho} | \exists t \in \Phi_{\mu\alpha\rho}(\mathcal{M}).r = C_{\mu\alpha\rho}t\} \\ M_E(\mathcal{M}) & := & \{r \in \Lambda_{\mu\alpha\rho} | \forall m \in \forall\mathcal{P}\forall\mathcal{Q}.(\mathcal{P}\to\mathcal{Q})\to\Phi(\mathcal{P})\to\Phi(\mathcal{Q})\forall\sigma\forall\mathcal{N}\in\mathsf{SAT}_\sigma \\ & & \forall s \in \Phi_\sigma(\mathcal{N})\to\mathcal{N}\, r\mathrm{E}_\mu\sigma ms \in \mathcal{N} \wedge \forall s \in \Phi_{\mu\alpha\rho\times\sigma}(\mathcal{M}\times\mathcal{N})\to\mathcal{N}\, r\mathrm{E}^+_\mu\sigma ms \in \mathcal{N}\} \end{array}$$

and define $\Psi_I : \mathsf{SAT}_{\mu\alpha\rho} \to \mathsf{SAT}_{\mu\alpha\rho}$ by

$$\Psi_I(\mathcal{M}) := \mathsf{cl}_{\mu\alpha\rho}(M_I(\mathcal{M}))$$

and $\Psi_E : \mathsf{SAT}_{\mu\alpha\rho} \to \mathsf{SAT}_{\mu\alpha\rho}$ by

$$\Psi_E(\mathcal{M}) := \mathsf{cl}_{\mu\alpha\rho}(M_E(\mathcal{M})).$$

We cannot ensure Ψ_I or Ψ_E to be monotone. In order to overcome this problem we define (with Pow denoting the power set)

$$\Phi^{\supseteq} : \mathsf{SAT}_{\mu\alpha\rho} \to \mathsf{Pow}(\mathsf{SN}_{\rho[\alpha:=\mu\alpha\rho]})$$

via

$$\begin{array}{rcl} \Phi^{\supseteq}(\mathcal{M}) & := & \{t \in \mathsf{SN}_{\rho[\alpha:=\mu\alpha\rho]} | \forall m \in \forall\mathcal{P}\forall\mathcal{Q}.(\mathcal{P}\to\mathcal{Q})\to\Phi(\mathcal{P})\to\Phi(\mathcal{Q}) \\ & & \forall\sigma\forall\mathcal{N}\in\mathsf{SAT}_\sigma\forall s\in\mathcal{M}\to\mathcal{N}.m(\mu\alpha\rho)\sigma st \in \Phi_\sigma(\mathcal{N})\} \end{array}$$

and for $m \in \forall\mathcal{P}\forall\mathcal{Q}.(\mathcal{P}\to\mathcal{Q})\to\Phi(\mathcal{P})\to\Phi(\mathcal{Q})$ we define $\Phi^{\subseteq}_m := (\Phi^{\subseteq}_{m,\sigma})_\sigma$ with

$$\Phi^{\subseteq}_{m,\sigma} : \mathsf{SAT}_\sigma \to \mathsf{SAT}_{\rho[\alpha:=\sigma]}$$

via

$$\begin{array}{rcl} \Phi^{\subseteq}_{m,\sigma}(\mathcal{N}) & := & \mathsf{cl}_{\rho[\alpha:=\sigma]}(\{r \in \Lambda_{\rho[\alpha:=\sigma]} | \exists\mathcal{M}\in\mathsf{SAT}_{\mu\alpha\rho}\exists s\in\mathcal{M}\to\mathcal{N}\exists t\in\Phi_{\mu\alpha\rho}(\mathcal{M}). \\ & & \qquad r = m(\mu\alpha\rho)\sigma st\}) \end{array}$$

$\Phi^{\supseteq}$ is monotone (even non-strictly positive) and $\Phi^{\supseteq} \supseteq \Phi_{\mu\alpha\rho}$ (pointwise) because of the rules ($\forall$-E) and ($\to$-E) for saturated sets. $\Phi^{\subseteq}_{m,\sigma}$ is monotone (even strictly positive) and $\Phi^{\subseteq}_{m,\sigma} \subseteq \Phi_\sigma$ (pointwise) also because of the rules ($\forall$-E) and ($\to$-E) for saturated sets. (Note that already the set which is the argument to $\mathsf{cl}_{\rho[\alpha:=\sigma]}$ is a subset of $\Phi_\sigma(\mathcal{M})$ and therefore also $\Phi^{\subseteq}_{m,\sigma}(\mathcal{M})$ because $\Phi_\sigma(\mathcal{M})$ is saturated.)
Define for $\mathcal{M} \in \mathsf{SAT}_{\mu\alpha\rho}$

$$\begin{array}{rcl} M^{\supseteq}_I(\mathcal{M}) & := & \{r \in \Lambda_{\mu\alpha\rho} | \exists t \in \Phi^{\supseteq}(\mathcal{M}).r = C_{\mu\alpha\rho}t\} \\ M^{\subseteq}_E(\mathcal{M}) & := & \{r \in \Lambda_{\mu\alpha\rho} | \forall m \in \forall\mathcal{P}\forall\mathcal{Q}.(\mathcal{P}\to\mathcal{Q})\to\Phi(\mathcal{P})\to\Phi(\mathcal{Q})\forall\sigma\forall\mathcal{N}\in\mathsf{SAT}_\sigma \\ & & \forall s \in \Phi_\sigma(\mathcal{N})\to\mathcal{N}\, r\mathrm{E}_\mu\sigma ms \in \mathcal{N} \wedge \forall s \in \Phi^{\subseteq}_{m,\mu\alpha\rho\times\sigma}(\mathcal{M}\times\mathcal{N})\to\mathcal{N}\, r\mathrm{E}^+_\mu\sigma ms \in \mathcal{N}\} \end{array}$$

and set

$$\mu_I(\Phi) := \text{ the least fixed point of } \Psi^{\supseteq}_I : \mathsf{SAT}_{\mu\alpha\rho} \to \mathsf{SAT}_{\mu\alpha\rho},$$

defined by

$$\Psi^{\supseteq}_I(\mathcal{M}) := \mathsf{cl}_{\mu\alpha\rho}(M^{\supseteq}_I(\mathcal{M}))$$

and

$$\mu_E(\Phi) := \text{ any fixed point of } \Psi^{\subseteq}_E : \mathsf{SAT}_{\mu\alpha\rho} \to \mathsf{SAT}_{\mu\alpha\rho},$$

defined by

$$\Psi^{\subseteq}_E(\mathcal{M}) := \mathsf{cl}_{\mu\alpha\rho}(M^{\subseteq}_E(\mathcal{M})).$$

Because Ψ_I and Ψ_E are monotone (they are even non-strictly positive) and $\mathsf{SAT}_{\mu\alpha\rho}$ is a complete lattice, such fixed points exists. Hence, $\mu_I(\Phi)$ and $\mu_E(\Phi)$ are $\mu\alpha\rho$-saturated sets.
Let us extract an encoding of types from the definition of $\mu_I(\Phi)$:

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

where we write $\mathsf{Mon}(\lambda\alpha\rho)$ for $\forall\alpha\forall\beta.(\alpha\to\beta)\to\rho\to\rho[\alpha:=\beta]$ and assume that $\beta \notin \{\alpha\}\cup\mathsf{FV}(\rho)$. The extraction read off the following proofs indeed gives an embedding of IMIT into NIPIT. Following the elimination-based proof one presumably gets an embedding into a system of non-interleaving positive fixed-point types.

Lemma A.17 $M_E(\mathcal{M}) \cap \mathsf{SN} \in \mathsf{SAT}$ and $M^{\subseteq}_E(\mathcal{M}) \cap \mathsf{SN} \in \mathsf{SAT}$ (which implies that $\Psi_E(\mathcal{M}) = M_E(\mathcal{M}) \cap \mathsf{SN}$ and $\Psi^{\subseteq}_E(\mathcal{M}) = M^{\subseteq}_E(\mathcal{M}) \cap \mathsf{SN}$).

Proof Straightforward. □

Lemma A.18 $M_I(\mathcal{M}) \subseteq \mathsf{SN}$ and $M_I^{\supseteq}(\mathcal{M}) \subseteq \mathsf{SN}$ (which implies that $\Psi_I(\mathcal{M}) \supseteq M_I(\mathcal{M})$ and $\Psi_I^{\supseteq}(\mathcal{M}) \supseteq M_I^{\supseteq}(\mathcal{M})$). □

Lemma A.19 Let $\mathcal{M}$ be a $\mu\alpha\rho$-saturated set $\mathcal{M}$.
$\mathcal{M}$ is a pre-fixed-point of Ψ_I iff for all $t \in \Phi_{\mu\alpha\rho}(\mathcal{M})$, we have that $C_{\mu\alpha\rho}t \in \mathcal{M}$.
$\mathcal{M}$ is a post-fixed-point of Ψ_E iff for all $r \in \mathcal{M}$, $m \in \forall\mathcal{P}\forall\mathcal{Q}.(\mathcal{P} \to \mathcal{Q}) \to \Phi(\mathcal{P}) \to \Phi(\mathcal{Q})$, types σ, $\mathcal{N} \in \mathsf{SAT}_\sigma$ and $s \in \Phi_\sigma(\mathcal{N}) \to \mathcal{N}$, we have $r\mathrm{E}_\mu\sigma ms \in \mathcal{N}$ and for all $s \in \Phi_{\mu\alpha\rho\times\sigma}(\mathcal{M} \times \mathcal{N}) \to \mathcal{N}$, $r\mathrm{E}_\mu^+\sigma ms \in \mathcal{N}$.

Proof Use the two preceding lemmas. □

Lemma A.20 $\mu_I(\Phi)$ is a pre-fixed-point of Ψ_I.

Proof $\mu_I(\Phi)$ is a pre-fixed-point of $\Psi_I^{\supseteq}$ by definition. Hence

$$\mu_I(\Phi) \supseteq \Psi_I^{\supseteq}(\mu_I(\Phi)) \supseteq \Psi_I(\mu_I(\Phi)),$$

which clearly follows from $\Phi^{\supseteq} \supseteq \Phi_{\mu\alpha\rho}$ because $M_I(\mathcal{M})$ and $M_I^{\supseteq}(\mathcal{M})$ have the same definition with exception of the occurrence of $\Phi_{\mu\alpha\rho}$ and $\Phi^{\supseteq}$, respectively, which is at a strictly positive position. □

Lemma A.21 $\mu_E(\Phi)$ is a post-fixed-point of Ψ_E.

Proof $\mu_E(\Phi)$ is a post-fixed-point of $\Psi_E^{\subseteq}$ by definition. Hence

$$\mu_E(\Phi) \subseteq \Psi_E^{\subseteq}(\mu_E(\Phi)) \subseteq \Psi_E(\mu_E(\Phi)),$$

which clearly follows from $\Phi_m^{\subseteq} \subseteq \Phi$ (typewise) because $M_E(\mathcal{M})$ and $M_E^{\subseteq}(\mathcal{M})$ have the same definition with exception of the occurrence of $\Phi_{\mu\alpha\rho\times\sigma}$ and $\Phi^{\subseteq}_{m,\mu\alpha\rho\times\sigma}$, respectively, which is at a non-strictly positive position. □

Lemma A.22 $\mu_E(\Phi)$ is a pre-fixed-point of Ψ_I[4].

Proof Set $\mathcal{M} := \mu_E(\Phi)$. We have to show that

$$\Psi_I(\mathcal{M}) \subseteq \mathcal{M}.$$

Because $\mathcal{M}$ is a pre-fixed-point of $\Psi_E^{\subseteq}$, it suffices to show

$$\Psi_I(\mathcal{M}) \subseteq \Psi_E^{\subseteq}(\mathcal{M}).$$

Because $\mathsf{cl}_{\mu\alpha\rho}$ is monotone, it suffices to show

$$M_I(\mathcal{M}) \subseteq M_E^{\subseteq}(\mathcal{M}).$$

Let $t \in \Phi_{\mu\alpha\rho}(\mathcal{M})$. We have to show

$$C_{\mu\alpha\rho}t \in M_E^{\subseteq}(\mathcal{M}).$$

[4]As for EMIT this seems not to extend to $\Psi_I^{\supseteq}$ instead of Ψ_I, and hence it is open whether $\mu_I(\Phi) \subseteq \mu_E(\Phi)$.

Let $m \in \forall\mathcal{P}\forall\mathcal{Q}.(\mathcal{P} \to \mathcal{Q}) \to \Phi(\mathcal{P}) \to \Phi(\mathcal{Q})$, σ be a type, $\mathcal{N} \in \mathsf{SAT}_\sigma$ and $s \in \Phi_\sigma(\mathcal{N}) \to \mathcal{N}$. We have to show that

$$C_{\mu\alpha\rho}t\mathrm{E}_\mu\sigma ms \in \mathcal{N}.$$

Because $\mathcal{N}$ is saturated it suffices to show

$$s\Big(m(\mu\alpha\rho)\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma ms)t\Big) \in \mathcal{N}.$$

Because of ($\to$-E) for saturated sets (in the sequel the named rules will always be those for saturated sets) it suffices to show

$$m(\mu\alpha\rho)\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma ms)t \in \Phi_\sigma(\mathcal{N}).$$

By ($\forall$-E) and ($\to$-E) it suffices to show

$$\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma ms \in \mathcal{M} \to \mathcal{N}.$$

We use ($\to$-I). Let $r \in \mathcal{M}$. We have to show

$$r\mathrm{E}_\mu\sigma ms \in \mathcal{N}.$$

This follows from $\mathcal{M} \subseteq \Psi_E^{\subseteq}(\mathcal{M}) \subseteq M_E^{\subseteq}(\mathcal{M})$.
We come to the case for which $\Psi_E^{\subseteq}$ was designed, namely $s \in \Phi^{\subseteq}_{m,\mu\alpha\rho\times\sigma}(\mathcal{M} \times \mathcal{N}) \to \mathcal{N}$. We have to show

$$C_{\mu\alpha\rho}t\mathrm{E}_\mu^+\sigma ms \in \mathcal{N}.$$

By saturatedness of $\mathcal{N}$ it suffices to show

$$s\Big(m(\mu\alpha\rho)(\mu\alpha\rho \times \sigma)\Big(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma ms)x\rangle\Big)t\Big) \in \mathcal{N}.$$

Because of ($\to$-E) it suffices to show

$$m(\mu\alpha\rho)(\mu\alpha\rho \times \sigma)\Big(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma ms)x\rangle\Big)t \in \Phi^{\subseteq}_{m,\mu\alpha\rho\times\sigma}(\mathcal{M} \times \mathcal{N}).$$

According to the definition of $\Phi_m^{\subseteq}$ (the closure in the definition of $\Phi_m^{\subseteq}$ only makes the set larger) we are done if we show

$$\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma ms)x\rangle \in \mathcal{M} \to \mathcal{M} \times \mathcal{N}.$$

We use ($\to$-I). Let $r \in \mathcal{M}$. We have to show

$$\langle r, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma ms)r\rangle \in \mathcal{M} \times \mathcal{N}.$$

By ($\times$-I) it suffices to show

$$(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma ms)r \in \mathcal{N}.$$

Because $\mathcal{N}$ is saturated, we only need to show

$$r\mathrm{E}_\mu^+\sigma ms \in \mathcal{N}.$$

This follows again from $\mathcal{M} \subseteq \Psi_E^{\subseteq}(\mathcal{M}) \subseteq M_E^{\subseteq}(\mathcal{M})$. □

Note that we used from $\mu_E(\Phi)$ the properties of being a pre-fixed-point and a post-fixed-point of $\Psi_E^{\subseteq}$.

Lemma A.23 $\mu_I(\Phi)$ is a post-fixed-point of Ψ_E[5].

Proof Set $\mathcal{M} := \mu_I(\Phi)$. We prove $\mathcal{M} \subseteq \Psi_E(\mathcal{M})$ by extended induction (as defined in the introduction to chapter 4) on $\mathcal{M}$. Let $\mathcal{M}' := \Psi_E(\mathcal{M}) \cap \mathcal{M}$. We have to show

$$\Psi_I^{\supseteq}(\mathcal{M}') \subseteq \Psi_E(\mathcal{M}).$$

Because $\mathsf{cl}_{\mu\alpha\rho}$ is monotone, it suffices to show

$$M_I^{\supseteq}(\mathcal{M}') \subseteq M_E(\mathcal{M}).$$

Let $r \in M_I^{\supseteq}(\mathcal{M}')$, i. e., $r = C_{\mu\alpha\rho}t$ with $t \in \Phi^{\supseteq}(\mathcal{M}')$. We have to show that

$$C_{\mu\alpha\rho}t \in M_E(\mathcal{M}).$$

Let $m \in \forall\mathcal{P}\forall\mathcal{Q}.(\mathcal{P} \to \mathcal{Q}) \to \Phi(\mathcal{P}) \to \Phi(\mathcal{Q})$, σ be a type, $\mathcal{N} \in \mathsf{SAT}_\sigma$ and $s \in \Phi_\sigma(\mathcal{N}) \to \mathcal{N}$. We have to show that

$$C_{\mu\alpha\rho}t\mathrm{E}_\mu\sigma ms \in \mathcal{N}.$$

Because $\mathcal{N}$ is saturated it suffices to show

$$s\Big(m(\mu\alpha\rho)\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma ms)t\Big) \in \mathcal{N}.$$

Because of ($\to$-E) for saturated sets it suffices to show

$$m(\mu\alpha\rho)\sigma(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma ms)t \in \Phi_\sigma(\mathcal{N}).$$

By definition of $\Phi^{\supseteq}(\mathcal{M}')$ it suffices to show

$$\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma ms \in \mathcal{M}' \to \mathcal{N}.$$

We use ($\to$-I). Let $r' \in \mathcal{M}'$. We have to show

$$r'\mathrm{E}_\mu\sigma ms \in \mathcal{N}.$$

This follows from $\mathcal{M}' \subseteq \Psi_E(\mathcal{M}) \subseteq M_E(\mathcal{M})$.
We come to the case which needs the extended form of induction. Let $s \in \Phi_{\mu\alpha\rho\times\sigma}(\mathcal{M}\times\mathcal{N}) \to \mathcal{N}$. We have to show

$$C_{\mu\alpha\rho}t\mathrm{E}^+_\mu\sigma ms \in \mathcal{N}.$$

By saturatedness of $\mathcal{N}$ it suffices to show

$$s\Big(m(\mu\alpha\rho)(\mu\alpha\rho\times\sigma)\Big(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma ms)x\rangle\Big)t\Big) \in \mathcal{N}.$$

Because of ($\to$-E) it suffices to show

$$m(\mu\alpha\rho)(\mu\alpha\rho\times\sigma)\Big(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma ms)x\rangle\Big)t \in \Phi_{\mu\alpha\rho\times\sigma}(\mathcal{M}\times\mathcal{N}).$$

By definition of $\Phi^{\supseteq}(\mathcal{M}')$ it suffices to show

$$\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma ms)x\rangle \in \mathcal{M}' \to \mathcal{M}\times\mathcal{N}.$$

[5]Again as for EMITthis seems not to extend to $\Psi_E^{\subseteq}$ instead of Ψ_E.

We use ($\to$-I). Let $r' \in \mathcal{M}'$. We have to show

$$\langle r', (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma ms)r'\rangle \in \mathcal{M}\times\mathcal{N}.$$

By ($\times$-I) it suffices to show

$$r' \in \mathcal{M} \text{ and } (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma ms)r' \in \mathcal{N}.$$

Because $\mathcal{N}$ is saturated and $\mathcal{M}' \subseteq \mathcal{M}$ (this argument needs the extension of induction), we only need to show

$$r'\mathrm{E}^+_\mu\sigma ms \in \mathcal{N}.$$

This follows again from $\mathcal{M}' \subseteq \Psi_E(\mathcal{M}) \subseteq M_E(\mathcal{M})$. □

Note that we only used that $\mu_I(\Phi)$ is a least pre-fixed-point.
Putting everything together we get the following rules for reasoning with $\mu_I(\Phi)$ and $\mu_E(\Phi)$ where we simply write $\mu(\Phi)$ for both of them:

(SAT) $\mu(\Phi) \in \mathsf{SAT}_{\mu\alpha\rho}$.

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

(μ-E) If $r \in \mu(\Phi)$, $m \in \forall\mathcal{P}\forall\mathcal{Q}.(\mathcal{P} \to \mathcal{Q}) \to \Phi(\mathcal{P}) \to \Phi(\mathcal{Q})$, σ type, $\mathcal{N} \in \mathsf{SAT}_\sigma$ and $s \in \Phi_\sigma(\mathcal{N}) \to \mathcal{N}$, then $r\mathrm{E}_\mu\sigma ms \in \mathcal{N}$.

(μ-E$^+$) If $r \in \mu(\Phi)$, $m \in \forall\mathcal{P}\forall\mathcal{Q}.(\mathcal{P} \to \mathcal{Q}) \to \Phi(\mathcal{P}) \to \Phi(\mathcal{Q})$, σ type, $\mathcal{N} \in \mathsf{SAT}_\sigma$ and $s \in \Phi_{\mu\alpha\rho\times\sigma}(\mu(\Phi)\times\mathcal{N}) \to \mathcal{N}$, then $r\mathrm{E}^+_\mu\sigma ms \in \mathcal{N}$.

A.2.3 Computability predicates for IMIT

Extend (as for EMIT) the definition of $\mathsf{SC}_\rho[\vec\alpha := \vec{\mathcal{P}}]$ by the following clause:

(μ) $\mathsf{SC}_{\mu\alpha\rho}[\vec\alpha := \vec{\mathcal{P}}] := \mu(\Phi)$ with $\Phi = (\Phi_\sigma)_\sigma$, where $\Phi_\sigma : \mathsf{SAT}_\sigma \to \mathsf{SAT}_{\rho[\vec\alpha,\alpha:=\vec\sigma,\sigma]}$ is defined by $\Phi_\sigma(\mathcal{P}) := \mathsf{SC}_\rho[\vec\alpha, \alpha := \vec{\mathcal{P}}, \mathcal{P}]$ (with the same assumptions as in the definition of strong computability for universal types in order to guarantee type-correctness).

The definition will once again be written more intuitively as follows:

$$\mathsf{SC}_{\mu\alpha\rho}[\vec\alpha := \vec{\mathcal{P}}] := \mu\mathcal{P}.\mathsf{SC}_\rho[\vec\alpha, \alpha := \vec{\mathcal{P}}, \mathcal{P}].$$

It is obvious that coincidence and substitution lemma (Lemma 9.13 and Lemma 9.14) extend to IMIT.
Now we extend Lemma 9.16 to IMIT:

Lemma A.24 Let r^ρ be a term, $\vec x^{\vec\rho}$ a list of variables and $\vec s$ an equally long list of terms such that $\vec s \in \mathsf{SC}_{\vec\rho}[\vec\alpha := \vec{\mathcal{P}}^{\vec\sigma}]$ for some $\vec{\mathcal{P}} \subseteq \mathsf{SAT}$. Assume that

$$\forall x\forall\pi\forall\pi'.(x^\pi, x^{\pi'} \in \mathsf{FV}(r) \cup \vec x^{\vec\rho}) \wedge (\pi[\vec\alpha := \vec\sigma] = \pi'[\vec\alpha := \vec\sigma]) \Rightarrow \pi = \pi'.$$

Then $r[\vec\alpha := \vec\sigma][\vec x^{\vec\rho[\vec\alpha:=\vec\sigma]} := \vec s] \in \mathsf{SC}_\rho[\vec\alpha := \vec{\mathcal{P}}]$.

Proof Induction on r. Again abbreviate $r[\vec{\alpha} := \vec{\sigma}][\vec{x}^{\vec{\rho}[\vec{\alpha}:=\vec{\sigma}]} := \vec{s}]$ by r' (for any term r). The assumption in the statement will again be referenced to as "injectivity". We only have to consider the rules (μ-I), (μ-E) and (μ-E$^+$).
(μ-I) Case $C_{\mu\alpha\rho}t$. $(C_{\mu\alpha\rho}t)' = C_{\mu\alpha\rho[\vec{\alpha}:=\vec{\sigma}]}t'$. We may assume that $\alpha \notin \vec{\alpha}$. We have to show $C_{\mu\alpha\rho[\vec{\alpha}:=\vec{\sigma}]}t' \in \mathsf{SC}_{\mu\alpha\rho}[\vec{\alpha} := \vec{\mathcal{P}}]$. By the rule ($\mu$-I) for saturated sets it suffices to show

$$t' \in \mathsf{SC}_\rho[\vec{\alpha}, \alpha := \vec{\mathcal{P}}, \mathsf{SC}_{\mu\alpha\rho}[\vec{\alpha} := \vec{\mathcal{P}}]].$$

Because $\mathsf{FV}(C_{\mu\alpha\rho}t) = \mathsf{FV}(t)$, injectivity holds trivially also for t. By induction hypothesis $t' \in \mathsf{SC}_{\rho[\alpha:=\mu\alpha\rho]}[\vec{\alpha} := \vec{\mathcal{P}}]$.
By the substitution lemma (which for system F is Lemma 9.14) we have

$$\mathsf{SC}_{\rho[\alpha:=\mu\alpha\rho]}[\vec{\alpha} := \vec{\mathcal{P}}] = \mathsf{SC}_\rho[\vec{\alpha}, \alpha := \vec{\mathcal{P}}, \mathsf{SC}_{\mu\alpha\rho}[\vec{\alpha} := \vec{\mathcal{P}}]].$$

(μ-E) Case $r\mathrm{E}_\mu\sigma s$. This is only a minor variant to the following (μ-E$^+$) and therefore not shown.
(μ-E$^+$) Case $r\mathrm{E}^+_\mu\sigma ms$. $(r\mathrm{E}^+_\mu\sigma ms)' = r'\mathrm{E}^+_\mu(\sigma[\vec{\alpha} := \vec{\sigma}])m's'$. We have to show that

$$r'\mathrm{E}^+_\mu\sigma[\vec{\alpha} := \vec{\sigma}]m's' \in \mathsf{SC}_\sigma[\vec{\alpha} := \vec{\mathcal{P}}].$$

Use (μ-E$^+$) for saturated sets. Show that $r' \in \mathsf{SC}_{\mu\alpha\rho}[\vec{\alpha} := \vec{\mathcal{P}}]$,

$$m' \in \forall\mathcal{P}\forall\mathcal{Q}.(\mathcal{P} \to \mathcal{Q}) \to \mathsf{SC}_\rho[\vec{\alpha}, \alpha := \vec{\mathcal{P}}, \mathcal{P}] \to \mathsf{SC}_\rho[\vec{\alpha}, \alpha := \vec{\mathcal{P}}, \mathcal{Q}] \text{ and}$$

$$s' \in \mathsf{SC}_\rho[\vec{\alpha}, \alpha := \vec{\mathcal{P}}, \mathsf{SC}_{\mu\alpha\rho}[\vec{\alpha} := \vec{\mathcal{P}}] \times \mathsf{SC}_\sigma[\vec{\alpha} := \vec{\mathcal{P}}]] \to \mathsf{SC}_\sigma[\vec{\alpha} := \vec{\mathcal{P}}].$$

Injectivity holds for r, m and s because $\mathsf{FV}(r), \mathsf{FV}(m), \mathsf{FV}(s) \subseteq \mathsf{FV}(r\mathrm{E}^+_\mu\sigma ms)$. Therefore by induction hypothesis we have $r' \in \mathsf{SC}_{\mu\alpha\rho}[\vec{\alpha} := \vec{\mathcal{P}}]$, $m' \in \mathsf{SC}_{\forall\alpha\forall\beta.(\alpha\to\beta)\to\rho\to\rho[\alpha:=\beta]}[\vec{\alpha} := \vec{\mathcal{P}}]$ and $s' \in \mathsf{SC}_{\rho[\alpha:=\mu\alpha\rho\times\sigma]\to\sigma}[\vec{\alpha} := \vec{\mathcal{P}}]$.
By two applications of the coincidence lemma (which for system F is Lemma 9.13) we see that

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

(one of the applications even needs the generalization which is in fact shown).
Because we may assume $\alpha \notin \vec{\alpha}$ the substitution lemma also gives the desired property of s'. □

As for EMIT we get as corollaries that $\mathsf{SN} = \Lambda$ and that every term is strongly normalizing.

A.3 Proof of strong normalization for varIMIT

This section contains the proof for $\mathsf{varIMIT}$ where the introduction-based proof is only a minor variant to the proof for IMIT. The elimination-based proof follows the more general approach to enforce monotonicity.
Extend the definition of SN for system eF by the expected clauses:

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

(β_μ) If $s\Big(m(\mu\alpha\rho)(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma ms)t\Big)\vec{s} \in \mathsf{SN}$, $x^{\mu\alpha\rho} \notin \mathsf{FV}(s)$ and $(C_{\mu\alpha\rho}t)\mathrm{E}_\mu\sigma ms\vec{s}$ is a term, then $(C_{\mu\alpha\rho}t)\mathrm{E}_\mu\sigma ms\vec{s} \in \mathsf{SN}$.

(β^+_μ) If $s\Big(m(\mu\alpha\rho)\Big(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma ms)x\rangle\Big)t\Big)\vec{s} \in \mathsf{SN}$, $x^{\mu\alpha\rho} \notin \mathsf{FV}(s)$ and the expression $(C_{\mu\alpha\rho}t)\mathrm{E}^+_\mu\sigma ms\vec{s}$ is a term, then $(C_{\mu\alpha\rho}t)\mathrm{E}^+_\mu\sigma ms\vec{s} \in \mathsf{SN}$.

Lemma A.25 If $t \in \mathsf{sn}$, then $C_{\mu\alpha\rho}t \in \mathsf{sn}$.

Proof Induction on $t \in \mathsf{sn}$. □

Lemma A.26 If $s\Big(m(\mu\alpha\rho)(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma ms)t\Big)\vec{s} \in \mathsf{sn}$, $x^{\mu\alpha\rho} \notin \mathsf{FV}(s)$ and $(C_{\mu\alpha\rho}t)\mathrm{E}_\mu\sigma ms\vec{s}$ is a term, then $(C_{\mu\alpha\rho}t)\mathrm{E}_\mu\sigma ms\vec{s} \in \mathsf{sn}$.

Proof Induction on $s\Big(m(\mu\alpha\rho)(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma ms)t\Big)\vec{s} \in \mathsf{sn}^+$ by the usual analysis of the reducts of $(C_{\mu\alpha\rho}t)\mathrm{E}_\mu\sigma ms\vec{s}$. (Note that we need sn^+ because s occurs twice in the canonical reduct of $(C_{\mu\alpha\rho}t)\mathrm{E}_\mu\sigma ms\vec{s}$.) □

Lemma A.27 If $s\Big(m(\mu\alpha\rho)\Big(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma ms)x\rangle\Big)t\Big)\vec{s} \in \mathsf{sn}$, $x^{\mu\alpha\rho} \notin \mathsf{FV}(s)$ and the expression $(C_{\mu\alpha\rho}t)\mathrm{E}^+_\mu\sigma ms\vec{s}$ is a term, then $(C_{\mu\alpha\rho}t)\mathrm{E}^+_\mu\sigma ms\vec{s} \in \mathsf{sn}$.

Proof Induction on $s\Big(m(\mu\alpha\rho)\Big(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma ms)x\rangle\Big)t\Big)\vec{s} \in \mathsf{sn}^+$ by analysis of the reducts of $(C_{\mu\alpha\rho}t)\mathrm{E}^+_\mu\sigma ms\vec{s}$. (Note that we need sn^+ because s occurs twice in the canonical reduct of $(C_{\mu\alpha\rho}t)\mathrm{E}^+_\mu\sigma ms\vec{s}$.) □

Corollary A.28 $\mathsf{SN} \subseteq \mathsf{sn}$.

Proof The same as for Lemma 9.6. □

A.3.1 Saturated sets for varIMIT

Extend the definition for system eF of $\mathcal{M}$ being a τ-saturated set by the following clause:

(β_μ) If $s\Big(m(\mu\alpha\rho)(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma ms)t\Big)\vec{s} \in \mathcal{M}$, $x^{\mu\alpha\rho} \notin \mathsf{FV}(s)$ and $(C_{\mu\alpha\rho}t)\mathrm{E}_\mu\sigma ms\vec{s}$ is a term, then $(C_{\mu\alpha\rho}t)\mathrm{E}_\mu\sigma ms\vec{s} \in \mathcal{M}$.

(β^+_μ) If $s\Big(m(\mu\alpha\rho)\Big(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma ms)x\rangle\Big)t\Big)\vec{s} \in \mathcal{M}$, $x^{\mu\alpha\rho} \notin \mathsf{FV}(s)$ and the expression $(C_{\mu\alpha\rho}t)\mathrm{E}^+_\mu\sigma ms\vec{s}$ is a term, then $(C_{\mu\alpha\rho}t)\mathrm{E}^+_\mu\sigma ms\vec{s} \in \mathcal{M}$.

Extend the definition of the τ-saturated closure $\mathsf{cl}_\tau(M)$ accordingly by the clause

(β_μ) If $s\Big(m(\mu\alpha\rho)(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma ms)t\Big)\vec{s} \in \mathsf{cl}_\tau(M)$, $x^{\mu\alpha\rho} \notin \mathsf{FV}(s)$ and $(C_{\mu\alpha\rho}t)\mathrm{E}_\mu\sigma ms\vec{s}$ is a term, then $(C_{\mu\alpha\rho}t)\mathrm{E}_\mu\sigma ms\vec{s} \in \mathsf{cl}_\tau(M)$.

(β^+_μ) If $s\Big(m(\mu\alpha\rho)\Big(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\sigma ms)x\rangle\Big)t\Big)\vec{s} \in \mathsf{cl}_\tau(M)$, $x^{\mu\alpha\rho} \notin \mathsf{FV}(s)$ and the expression $(C_{\mu\alpha\rho}t)\mathrm{E}^+_\mu\sigma ms\vec{s}$ is a term, then $(C_{\mu\alpha\rho}t)\mathrm{E}^+_\mu\sigma ms\vec{s} \in \mathsf{cl}_\tau(M)$.

Again $\mathsf{cl}_\tau(M)$ is the smallest τ-saturated set containing $M \cap \mathsf{SN}_\tau$.

A.3.2 Calculating with saturated sets for varIMIT

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

$$\begin{array}{rcl}
M_I(\mathcal{M}) & := & \{r \in \Lambda_{\mu\alpha\rho} | \exists t \in \Phi_{\mu\alpha\rho}(\mathcal{M}).r = C_{\mu\alpha\rho}t\} \\
M_E(\mathcal{M}) & := & \{r \in \Lambda_{\mu\alpha\rho} | \forall\sigma\forall\mathcal{N} \in \mathsf{SAT}_\sigma \\
 & & \forall m \in \forall\mathcal{P}.(\mathcal{P} \to \mathcal{N}) \to \Phi(\mathcal{P}) \to \Phi_\sigma(\mathcal{N}) \\
 & & \forall s \in \Phi_\sigma(\mathcal{N}) \to \mathcal{N}\, r\mathrm{E}_\mu\sigma ms \in \mathcal{N} \wedge \\
 & & \forall m \in \forall\mathcal{P}.(\mathcal{P} \to \mathcal{M} \times \mathcal{N}) \to \Phi(\mathcal{P}) \to \Phi_{\mu\alpha\rho\times\sigma}(\mathcal{M} \times \mathcal{N}) \\
 & & \forall s \in \Phi_{\mu\alpha\rho\times\sigma}(\mathcal{M} \times \mathcal{N}) \to \mathcal{N}\, r\mathrm{E}^+_\mu\sigma ms \in \mathcal{N}\}
\end{array}$$

and define $\Psi_I : \mathsf{SAT}_{\mu\alpha\rho} \to \mathsf{SAT}_{\mu\alpha\rho}$ by

$$\Psi_I(\mathcal{M}) := \mathsf{cl}_{\mu\alpha\rho}(M_I(\mathcal{M}))$$

and $\Psi_E : \mathsf{SAT}_{\mu\alpha\rho} \to \mathsf{SAT}_{\mu\alpha\rho}$ by

$$\Psi_E(\mathcal{M}) := \mathsf{cl}_{\mu\alpha\rho}(M_E(\mathcal{M})).$$

We cannot ensure Ψ_I or Ψ_E to be monotone. In order to overcome this problem we define (with Pow denoting the power set)

$$\Phi^{\supseteq} : \mathsf{SAT}_{\mu\alpha\rho} \to \mathsf{Pow}(\mathsf{SN}_{\rho[\alpha:=\mu\alpha\rho]})$$

via

$$\begin{array}{rcl} \Phi^{\supseteq}(\mathcal{M}) & := & \{t \in \mathsf{SN}_{\rho[\alpha:=\mu\alpha\rho]} | \forall\sigma\forall\mathcal{N} \in \mathsf{SAT}_\sigma \forall m \in \forall\mathcal{P}.(\mathcal{P} \to \mathcal{N}) \to \Phi(\mathcal{P}) \to \Phi_\sigma(\mathcal{N}) \\ & & \qquad \forall s \in \mathcal{M} \to \mathcal{N}.m(\mu\alpha\rho)st \in \Phi_\sigma(\mathcal{N})\} \end{array}$$

and $\Phi^{\subseteq} := (\Phi^{\subseteq}_\sigma)_\sigma$ with

$$\Phi^{\subseteq}_\sigma : \mathsf{SAT}_\sigma \to \mathsf{SAT}_{\rho[\alpha:=\sigma]}$$

via

$$\begin{array}{rcl} \Phi^{\subseteq}_\sigma(\mathcal{N}) & := & \mathsf{cl}_{\rho[\alpha:=\sigma]}(\{r \in \Lambda_{\rho[\alpha:=\sigma]} | \exists m \in \forall\mathcal{P}.(\mathcal{P} \to \mathcal{N}) \to \Phi(\mathcal{P}) \to \Phi_\sigma(\mathcal{N}) \exists \mathcal{M} \in \mathsf{SAT}_{\mu\alpha\rho} \\ & & \qquad \exists s \in \mathcal{M} \to \mathcal{N} \exists t \in \Phi_{\mu\alpha\rho}(\mathcal{M}).r = m(\mu\alpha\rho)st\}) \end{array}$$

$\Phi^{\supseteq}$ is monotone (even non-strictly positive) and $\Phi^{\supseteq} \supseteq \Phi_{\mu\alpha\rho}$ (pointwise) because of the rules ($\forall$-E) and ($\to$-E) for saturated sets. $\Phi^{\subseteq}_\sigma$ is obviously not monotone (in general) and $\Phi^{\subseteq}_\sigma \subseteq \Phi_\sigma$ (pointwise) also because of the rules ($\forall$-E) and ($\to$-E) for saturated sets. (Note that already the set which is the argument to $\mathsf{cl}_{\rho[\alpha:=\sigma]}$ is a subset of $\Phi_\sigma(\mathcal{M})$ and therefore also $\Phi^{\subseteq}_\sigma(\mathcal{M})$ because $\Phi_\sigma(\mathcal{M})$ is saturated.)
Define for $\mathcal{M} \in \mathsf{SAT}_{\mu\alpha\rho}$

$$\begin{array}{rcl} M^{\supseteq}_I(\mathcal{M}) & := & \{r \in \Lambda_{\mu\alpha\rho} | \exists t \in \Phi^{\supseteq}(\mathcal{M}).r = C_{\mu\alpha\rho}t\} \\ M^{\subseteq}_E(\mathcal{M}) & := & \{r \in \Lambda_{\mu\alpha\rho} | \forall\sigma\forall\mathcal{N} \in \mathsf{SAT}_\sigma \\ & & \forall m \in \forall\mathcal{P}.(\mathcal{P} \to \mathcal{N}) \to \Phi(\mathcal{P}) \to \Phi_\sigma(\mathcal{N}) \\ & & \forall s \in \Phi_\sigma(\mathcal{N}) \to \mathcal{N}\, r\mathrm{E}_\mu \sigma m s \in \mathcal{N} \wedge \\ & & \forall m \in \forall\mathcal{P}.(\mathcal{P} \to \mathcal{M} \times \mathcal{N}) \to \Phi(\mathcal{P}) \to \Phi_{\mu\alpha\rho\times\sigma}(\mathcal{M} \times \mathcal{N}) \\ & & \forall s \in \Phi^{\subseteq}_{\mu\alpha\rho\times\sigma}(\mathcal{M} \times \mathcal{N}) \to \mathcal{N}\, r\mathrm{E}^+_\mu \sigma m s \in \mathcal{N}\} \end{array}$$

and set

$$\mu_I(\Phi) := \text{ the least fixed point of } \Psi^{\supseteq}_I : \mathsf{SAT}_{\mu\alpha\rho} \to \mathsf{SAT}_{\mu\alpha\rho},$$

defined by

$$\Psi^{\supseteq}_I(\mathcal{M}) := \mathsf{cl}_{\mu\alpha\rho}(M^{\supseteq}_I(\mathcal{M}))$$

and define $\Psi^{\subseteq}_E : \mathsf{SAT}_{\mu\alpha\rho} \to \mathsf{SAT}_{\mu\alpha\rho}$ by

$$\Psi^{\subseteq}_E(\mathcal{M}) := \mathsf{cl}_{\mu\alpha\rho}(M^{\subseteq}_E(\mathcal{M})).$$

Because Ψ_I is monotone (it is even non-strictly positive) and $\mathsf{SAT}_{\mu\alpha\rho}$ is a complete lattice, such a fixed point exists. Hence, $\mu_I(\Phi)$ is a $\mu\alpha\rho$-saturated set. $\Psi^{\subseteq}_E$ cannot be monotone in general

(not only the problem with $\Phi^{\subseteq}$ propagates but also $M^{\subseteq}_E(\mathcal{M})$'s definition has several critical occurrences of $\mathcal{M}$). The definitions of $\Phi^{\subseteq}$, $M^{\subseteq}_E(\mathcal{M})$ and $\Psi^{\subseteq}_E$ are thus useless. $\Phi^{\supseteq}$ will not be used any further and $M^{\subseteq}_E(\mathcal{M})$ and $\Psi^{\subseteq}_E$ are redefined as follows:

$$\begin{array}{rcl} M^{\subseteq}_E(\mathcal{M}) & := & \{r \in \Lambda_{\mu\alpha\rho} | \forall\sigma\forall\mathcal{N} \in \mathsf{SAT}_\sigma \\ & & \forall m \in \forall\mathcal{P}.(\mathcal{P} \to \mathcal{N}) \to \Phi(\mathcal{P}) \to \Phi_\sigma(\mathcal{N}) \\ & & \forall s \in \Phi_\sigma(\mathcal{N}) \to \mathcal{N}\, r\mathrm{E}_\mu \sigma m s \in \mathcal{N} \wedge \\ & & \forall \mathcal{M}' \in \mathsf{SAT}_{\mu\alpha\rho}.\mathcal{M} \subseteq \mathcal{M}' \Rightarrow \\ & & \forall m \in \forall\mathcal{P}.(\mathcal{P} \to \mathcal{M}' \times \mathcal{N}) \to \Phi(\mathcal{P}) \to \Phi_{\mu\alpha\rho\times\sigma}(\mathcal{M}' \times \mathcal{N}) \\ & & \forall s \in \Phi_{\mu\alpha\rho\times\sigma}(\mathcal{M}' \times \mathcal{N}) \to \mathcal{N}\, r\mathrm{E}^+_\mu \sigma m s \in \mathcal{N}\} \\ \Psi^{\subseteq}_E(\mathcal{M}) & := & \mathsf{cl}_{\mu\alpha\rho}(M^{\subseteq}_E(\mathcal{M})) \end{array}$$

$\Psi^{\subseteq}_E$ is monotone (even non-strictly positive) and obviously $\Psi^{\subseteq}_E \subseteq \Psi_E$ (pointwise). We may now form

$$\mu_E(\Phi) := \text{ any fixed point of } \Psi^{\subseteq}_E,$$

which therefore is also a post-fixed-point of Ψ_E.
An analysis of the definition of $\mu_I(\Phi)$ gives the following encoding (for "new" β and γ):

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

An embedding of $\mathsf{varIMIT}$ into NIPIT may now be read off the proofs that $\mu_I(\Phi)$ is a pre-fixed-point of Ψ_I and that $\mu_I(\Phi)$ is a post-fixed-point of Ψ_E[6] which are given below. It is possible to show that this indeed gives an embedding.
The elimination-based approach presumably gives a quite complicated embedding into non-interleaving positive fixed-point types.

Lemma A.29 $M_E(\mathcal{M}) \cap \mathsf{SN} \in \mathsf{SAT}$ and $M^{\subseteq}_E(\mathcal{M}) \cap \mathsf{SN} \in \mathsf{SAT}$ (which implies that $\Psi_E(\mathcal{M}) = M_E(\mathcal{M}) \cap \mathsf{SN}$ and $\Psi^{\subseteq}_E(\mathcal{M}) = M^{\subseteq}_E(\mathcal{M}) \cap \mathsf{SN}$).

Proof Straightforward. □

Lemma A.30 $M_I(\mathcal{M}) \subseteq \mathsf{SN}$ and $M^{\supseteq}_I(\mathcal{M}) \subseteq \mathsf{SN}$ (which implies that $\Psi_I(\mathcal{M}) \supseteq M_I(\mathcal{M})$ and $\Psi^{\supseteq}_I(\mathcal{M}) \supseteq M^{\supseteq}_I(\mathcal{M})$). □

Lemma A.31 Let $\mathcal{M}$ be a $\mu\alpha\rho$-saturated set $\mathcal{M}$.
$\mathcal{M}$ is a pre-fixed-point of Ψ_I iff for all $t \in \Phi_{\mu\alpha\rho}(\mathcal{M})$, we have that $C_{\mu\alpha\rho}t \in \mathcal{M}$.
$\mathcal{M}$ is a post-fixed-point of Ψ_E iff for all $r \in \mathcal{M}$, types σ, $\mathcal{N} \in \mathsf{SAT}_\sigma$, $m \in \forall\mathcal{P}.(\mathcal{P} \to \mathcal{N}) \to \Phi(\mathcal{P}) \to \Phi_\sigma(\mathcal{N})$ and $s \in \Phi_\sigma(\mathcal{N}) \to \mathcal{N}$, we have $r\mathrm{E}_\mu\sigma ms \in \mathcal{N}$ and for $m \in \forall\mathcal{P}.(\mathcal{P} \to \mathcal{M} \times \mathcal{N}) \to \Phi(\mathcal{P}) \to \Phi_{\mu\alpha\rho\times\sigma}(\mathcal{M} \times \mathcal{N})$ and $s \in \Phi_{\mu\alpha\rho\times\sigma}(\mathcal{M} \times \mathcal{N}) \to \mathcal{N}$, $r\mathrm{E}^+_\mu\sigma ms \in \mathcal{N}$.

Proof Use the two preceding lemmas. □

Lemma A.32 Every pre-fixed-point of $\Psi^{\supseteq}_I$ is also a pre-fixed-point of Ψ_I.

Proof Let $\mathcal{M}$ be a pre-fixed-point of $\Psi^{\supseteq}_I$. Hence

$$\mathcal{M} \supseteq \Psi^{\supseteq}_I(\mathcal{M}) \supseteq \Psi_I(\mathcal{M}),$$

which clearly follows from $\Phi^{\supseteq} \supseteq \Phi_{\mu\alpha\rho}$ because $M_I(\mathcal{M})$ and $M^{\supseteq}_I(\mathcal{M})$ have the same definition with exception of the occurrence of $\Phi_{\mu\alpha\rho}$ and $\Phi^{\supseteq}$, respectively, which is at a strictly positive position. □

[6] In fact we prove that $\mu_I(\Phi)$ is a post-fixed-point of $\Psi^{\subseteq}_E$ and should analyze the slightly easier proof of the weaker statement.

Lemma A.33 $\mu_E(\Phi)$ is a pre-fixed-point of $\Psi_I^{\supseteq}$.

Proof Set $\mathcal{M} := \mu_E(\Phi)$. We have to show that

$$\Psi_I^{\supseteq}(\mathcal{M}) \subseteq \mathcal{M}.$$

Because $\mathcal{M}$ is a pre-fixed-point of $\Psi_E^{\subseteq}$, it suffices to show

$$\Psi_I^{\supseteq}(\mathcal{M}) \subseteq \Psi_E^{\subseteq}(\mathcal{M}).$$

Because $\mathsf{cl}_{\mu\alpha\rho}$ is monotone, it suffices to show

$$M_I^{\supseteq}(\mathcal{M}) \subseteq M_E^{\subseteq}(\mathcal{M}).$$

Let $t \in \Phi^{\supseteq}(\mathcal{M})$. We have to show

$$C_{\mu\alpha\rho}t \in M_E^{\subseteq}(\mathcal{M}).$$

Let σ be a type, $\mathcal{N} \in \mathsf{SAT}_\sigma$, $m \in \forall\mathcal{P}.(\mathcal{P} \to \mathcal{N}) \to \Phi(\mathcal{P}) \to \Phi_\sigma(\mathcal{N})$ and $s \in \Phi_\sigma(\mathcal{N}) \to \mathcal{N}$. We have to show that

$$C_{\mu\alpha\rho}t\mathrm{E}_\mu\sigma ms \in \mathcal{N}.$$

Because $\mathcal{N}$ is saturated it suffices to show

$$s\Big(m(\mu\alpha\rho)(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma ms)t\Big) \in \mathcal{N}.$$

Because of ($\to$-E) for saturated sets (in the sequel the named rules will always be those for saturated sets) it suffices to show

$$m(\mu\alpha\rho)(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma ms)t \in \Phi_\sigma(\mathcal{N}).$$

By definition of $\Phi^{\supseteq}(\mathcal{M})$ it suffices to show

$$\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma ms \in \mathcal{M} \to \mathcal{N}.$$

We use ($\to$-I). Let $r \in \mathcal{M}$. We have to show

$$r\mathrm{E}_\mu\sigma ms \in \mathcal{N}.$$

This follows from $\mathcal{M} \subseteq \Psi_E^{\subseteq}(\mathcal{M}) \subseteq M_E^{\subseteq}(\mathcal{M})$.
We come to the case for which $\Psi_E^{\subseteq}$ was designed, namely $m \in \forall\mathcal{P}.(\mathcal{P} \to \mathcal{M}' \times \mathcal{N}) \to \Phi(\mathcal{P}) \to \Phi_{\mu\alpha\rho\times\sigma}(\mathcal{M}' \times \mathcal{N})$ and $s \in \Phi_{\mu\alpha\rho\times\sigma}(\mathcal{M}' \times \mathcal{N}) \to \mathcal{N}$ for some $\mathcal{M}' \in \mathsf{SAT}_{\mu\alpha\rho}$ with $\mathcal{M} \subseteq \mathcal{M}'$. We have to show

$$C_{\mu\alpha\rho}t\mathrm{E}_\mu^+\sigma ms \in \mathcal{N}.$$

By saturatedness of $\mathcal{N}$ it suffices to show

$$s\Big(m(\mu\alpha\rho)\Big(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma ms)x\rangle\Big)t\Big) \in \mathcal{N}.$$

Because of ($\to$-E) it suffices to show

$$m(\mu\alpha\rho)(\mu\alpha\rho \times \sigma)\Big(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma ms)x\rangle\Big)t \in \Phi_{\mu\alpha\rho\times\sigma}(\mathcal{M}' \times \mathcal{N}).$$

According to the definition of $\Phi^{\supseteq}$ we are done if we show

$$\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma ms)x\rangle \in \mathcal{M} \to \mathcal{M}' \times \mathcal{N}.$$

We use ($\to$-I). Let $r \in \mathcal{M}$. We have to show

$$\langle r, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma ms)r\rangle \in \mathcal{M}' \times \mathcal{N}.$$

By ($\times$-I) and because of $\mathcal{M} \subseteq \mathcal{M}'$ it suffices to show

$$(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma ms)r \in \mathcal{N}.$$

Because $\mathcal{N}$ is saturated, we only need to show

$$r\mathrm{E}_\mu^+\sigma ms \in \mathcal{N}.$$

This follows again from $\mathcal{M} \subseteq \Psi_E^{\subseteq}(\mathcal{M}) \subseteq M_E^{\subseteq}(\mathcal{M})$. □

Note that we used from $\mu_E(\Phi)$ the properties of being a pre-fixed-point and a post-fixed-point of $\Psi_E^{\subseteq}$. Note also that for the purposes of the normalization proof it would have been enough to show that $\mu_E(\Phi)$ is a pre-fixed-point of Ψ_I instead of $\Psi_I^{\supseteq}$, but from the above result we even get the following

Corollary A.34 $\mu_I(\Phi) \subseteq \mu_E(\Phi)$.

Proof $\mu_I(\Phi)$ is the least pre-fixed-point of $\Psi_I^{\supseteq}$. □

Lemma A.35 $\mu_I(\Phi)$ is a post-fixed-point of $\Psi_E^{\subseteq}$.

Proof Set $\mathcal{M} := \mu_I(\Phi)$. We prove $\mathcal{M} \subseteq \Psi_E^{\subseteq}(\mathcal{M})$ by extended induction (as defined in the introduction to chapter 4) on $\mathcal{M}$. Let $\mathcal{M}' := \Psi_E^{\subseteq}(\mathcal{M}) \cap \mathcal{M}$. We have to show

$$\Psi_I^{\supseteq}(\mathcal{M}') \subseteq \Psi_E^{\subseteq}(\mathcal{M}).$$

Because $\mathsf{cl}_{\mu\alpha\rho}$ is monotone, it suffices to show

$$M_I^{\supseteq}(\mathcal{M}') \subseteq M_E^{\subseteq}(\mathcal{M}).$$

Let $r \in M_I^{\supseteq}(\mathcal{M}')$, i.e., $r = C_{\mu\alpha\rho}t$ with $t \in \Phi^{\supseteq}(\mathcal{M}')$. We have to show that

$$C_{\mu\alpha\rho}t \in M_E^{\subseteq}(\mathcal{M}).$$

Let σ be a type, $\mathcal{N} \in \mathsf{SAT}_\sigma$, $m \in \forall\mathcal{P}.(\mathcal{P} \to \mathcal{N}) \to \Phi(\mathcal{P}) \to \Phi_\sigma(\mathcal{N})$ and $s \in \Phi_\sigma(\mathcal{N}) \to \mathcal{N}$. We have to show that

$$C_{\mu\alpha\rho}t\mathrm{E}_\mu\sigma ms \in \mathcal{N}.$$

Because $\mathcal{N}$ is saturated it suffices to show

$$s\Big(m(\mu\alpha\rho)(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma ms)t\Big) \in \mathcal{N}.$$

Because of ($\to$-E) for saturated sets it suffices to show

$$m(\mu\alpha\rho)(\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma ms)t \in \Phi_\sigma(\mathcal{N}).$$

By definition of $\Phi^{\supseteq}(\mathcal{M}')$ it suffices to show

$$\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu\sigma ms \in \mathcal{M}' \to \mathcal{N}.$$

We use (→-I). Let $r' \in \mathcal{M}'$. We have to show

$$r'\mathrm{E}_\mu \sigma m s \in \mathcal{N}.$$

This follows from $\mathcal{M}' \subseteq \Psi_E^{\subseteq}(\mathcal{M}) \subseteq M_E^{\subseteq}(\mathcal{M})$.
We come to the case which needs the extended form of induction. Let $\mathcal{M}'' \in \mathsf{SAT}_{\mu\alpha\rho}$ with $\mathcal{M} \subseteq \mathcal{M}''$, $m \in \forall\mathcal{P}.(\mathcal{P} \to \mathcal{M}'' \times \mathcal{N}) \to \Phi(\mathcal{P}) \to \Phi_{\mu\alpha\rho\times\sigma}(\mathcal{M}'' \times \mathcal{N})$ and $s \in \Phi_{\mu\alpha\rho\times\sigma}(\mathcal{M}'' \times \mathcal{N}) \to \mathcal{N}$.
We have to show

$$C_{\mu\alpha\rho} t \mathrm{E}_\mu^+ \sigma m s \in \mathcal{N}.$$

By saturatedness of $\mathcal{N}$ it suffices to show

$$s\Big(m(\mu\alpha\rho)\Big(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma m s)x\rangle\Big)t\Big) \in \mathcal{N}.$$

Because of (→-E) it suffices to show

$$m(\mu\alpha\rho)\Big(\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma m s)x\rangle\Big)t \in \Phi_{\mu\alpha\rho\times\sigma}(\mathcal{M}'' \times \mathcal{N}).$$

By definition of $\Phi^{\supseteq}(\mathcal{M}')$ it suffices to show

$$\lambda x^{\mu\alpha\rho}.\langle x, (\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma m s)x\rangle \in \mathcal{M}' \to \mathcal{M}'' \times \mathcal{N}.$$

We use (→-I). Let $r' \in \mathcal{M}'$. We have to show

$$\langle r', (\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma m s)r'\rangle \in \mathcal{M}'' \times \mathcal{N}.$$

By (×-I) it suffices to show

$$r' \in \mathcal{M}'' \text{ and } (\lambda x^{\mu\alpha\rho}.x\mathrm{E}_\mu^+\sigma m s)r' \in \mathcal{N}.$$

Because $\mathcal{N}$ is saturated and $\mathcal{M}' \subseteq \mathcal{M} \subseteq \mathcal{M}''$ (this argument needs the extension of induction), we only need to show

$$r'\mathrm{E}_\mu^+ \sigma m s \in \mathcal{N}.$$

This follows again from $\mathcal{M}' \subseteq \Psi_E^{\subseteq}(\mathcal{M}) \subseteq M_E^{\subseteq}(\mathcal{M})$. □

Note that we only used that $\mu_I(\Phi)$ is a least pre-fixed-point. Note also that for the purposes of the normalization proof it would have been enough to show that $\mu_I(\Phi)$ is a post-fixed-point of Ψ_E instead of $\Psi_E^{\subseteq}$[7].
Putting everything together we get the following rules for reasoning with $\mu(\Phi) := \mu_I(\Phi)$:

- (SAT) $\mu(\Phi) \in \mathsf{SAT}_{\mu\alpha\rho}$.
- (μ-I) If $t \in \Phi_{\mu\alpha\rho}(\mu(\Phi))$, then $C_{\mu\alpha\rho} t \in \mu(\Phi)$.
- (μ-E) If $r \in \mu(\Phi)$, σ type, $\mathcal{N} \in \mathsf{SAT}_\sigma$, $m \in \forall\mathcal{P}.(\mathcal{P} \to \mathcal{N}) \to \Phi(\mathcal{P}) \to \Phi_\sigma(\mathcal{N})$ and $s \in \Phi_\sigma(\mathcal{N}) \to \mathcal{N}$, then $r\mathrm{E}_\mu\sigma m s \in \mathcal{N}$.
- (μ-E$^+$) If $r \in \mu(\Phi)$, σ type, $\mathcal{N} \in \mathsf{SAT}_\sigma$, $m \in \forall\mathcal{P}.(\mathcal{P} \to \mathcal{M}\times\mathcal{N}) \to \Phi(\mathcal{P}) \to \Phi_{\mu\alpha\rho\times\sigma}(\mathcal{M}\times\mathcal{N})$ and $s \in \Phi_{\mu\alpha\rho\times\sigma}(\mu(\Phi) \times \mathcal{N}) \to \mathcal{N}$, then $r\mathrm{E}_\mu^+\sigma m s \in \mathcal{N}$.

[7] If the greatest fixed point is taken for $\mu_E(\Phi)$, then also the preceding lemma implies that $\mu_I(\Phi) \subseteq \mu_E(\Phi)$.

A.3.3 Computability predicates for varIMIT

Extend (as for IMIT) the definition of $\mathsf{SC}_\rho[\vec\alpha := \vec{\mathcal{P}}]$ by the following clause:

(μ) $\mathsf{SC}_{\mu\alpha\rho}[\vec\alpha := \vec{\mathcal{P}}] := \mu(\Phi)$ with $\Phi = (\Phi_\sigma)_\sigma$, where $\Phi_\sigma : \mathsf{SAT}_\sigma \to \mathsf{SAT}_{\rho[\vec\alpha,\alpha:=\vec\sigma,\sigma]}$ is defined by $\Phi_\sigma(\mathcal{P}) := \mathsf{SC}_\rho[\vec\alpha, \alpha := \vec{\mathcal{P}}, \mathcal{P}]$ (with the same assumptions as in the definition of strong computability for universal types in order to guarantee type-correctness).

The definition will once again be written more intuitively as follows:

$$\mathsf{SC}_{\mu\alpha\rho}[\vec\alpha := \vec{\mathcal{P}}] := \mu\mathcal{P}.\mathsf{SC}_\rho[\vec\alpha, \alpha := \vec{\mathcal{P}}, \mathcal{P}].$$

It is obvious that coincidence and substitution lemma (Lemma 9.13 and Lemma 9.14) extend to varIMIT.
Now we extend Lemma 9.16 to varIMIT:

Lemma A.36 Let r^ρ be a term, $\vec x^{\vec\rho}$ a list of variables and $\vec s$ an equally long list of terms such that $\vec s \in \mathsf{SC}_{\vec\rho}[\vec\alpha := \vec{\mathcal{P}}^{\vec\sigma}]$ for some $\vec{\mathcal{P}} \subseteq \mathsf{SAT}$. Assume that

$$\forall x \forall \pi \forall \pi'.(x^\pi, x^{\pi'} \in \mathsf{FV}(r) \cup \vec x^{\vec\rho}) \wedge (\pi[\vec\alpha := \vec\sigma] = \pi'[\vec\alpha := \vec\sigma]) \Rightarrow \pi = \pi'.$$

Then $r[\vec\alpha := \vec\sigma][\vec x^{\vec\rho[\vec\alpha:=\vec\sigma]} := \vec s] \in \mathsf{SC}_\rho[\vec\alpha := \vec{\mathcal{P}}]$.

Proof Induction on r. Again abbreviate $r[\vec\alpha := \vec\sigma][\vec x^{\vec\rho[\vec\alpha:=\vec\sigma]} := \vec s]$ by r' (for any term r). The assumption in the statement will again be referenced to as "injectivity". We only have to consider the rules (μ-I), (μ-E) and (μ-E$^+$).
(μ-I) Trivially the same as for IMIT.
(μ-E) Case $r\mathrm{E}_\mu\sigma s$. This is only a minor variant to the following (μ-E$^+$) and therefore not shown.
(μ-E$^+$) Case $r\mathrm{E}_\mu^+\sigma m s$. $(r\mathrm{E}_\mu^+\sigma m s)' = r'\mathrm{E}_\mu^+(\sigma[\vec\alpha := \vec\sigma])m's'$. We have to show that

$$r'\mathrm{E}_\mu^+\sigma[\vec\alpha := \vec\sigma]m's' \in \mathsf{SC}_\sigma[\vec\alpha := \vec{\mathcal{P}}].$$

Use (μ-E$^+$) for saturated sets. Show that $r' \in \mathsf{SC}_{\mu\alpha\rho}[\vec\alpha := \vec{\mathcal{P}}]$,

$$m' \in \forall\mathcal{P}.(\mathcal{P} \to \mathsf{SC}_{\mu\alpha\rho}[\vec\alpha := \vec{\mathcal{P}}] \times \mathsf{SC}_\sigma[\vec\alpha := \vec{\mathcal{P}}]) \to \mathsf{SC}_\rho[\vec\alpha, \alpha := \vec{\mathcal{P}}, \mathcal{P}] \to \mathsf{SC}_\rho[\vec\alpha, \alpha := \vec{\mathcal{P}}, \mathsf{SC}_{\mu\alpha\rho}[\vec\alpha := \vec{\mathcal{P}}] \times \mathsf{SC}_\sigma[\vec\alpha := \vec{\mathcal{P}}]]$$

and

$$s' \in \mathsf{SC}_\rho[\vec\alpha, \alpha := \vec{\mathcal{P}}, \mathsf{SC}_{\mu\alpha\rho}[\vec\alpha := \vec{\mathcal{P}}] \times \mathsf{SC}_\sigma[\vec\alpha := \vec{\mathcal{P}}]] \to \mathsf{SC}_\sigma[\vec\alpha := \vec{\mathcal{P}}].$$

Injectivity holds for r, m and s because $\mathsf{FV}(r), \mathsf{FV}(m), \mathsf{FV}(s) \subseteq \mathsf{FV}(r\mathrm{E}_\mu^+\sigma m s)$. Therefore by induction hypothesis we have $r' \in \mathsf{SC}_{\mu\alpha\rho}[\vec\alpha := \vec{\mathcal{P}}]$, $m' \in \mathsf{SC}_{\forall\alpha.(\alpha\to\mu\alpha\rho\times\sigma)\to\rho\to\rho[\alpha:=\mu\alpha\rho\times\sigma]}[\vec\alpha := \vec{\mathcal{P}}]$ and $s' \in \mathsf{SC}_{\rho[\alpha:=\mu\alpha\rho\times\sigma]\to\sigma}[\vec\alpha := \vec{\mathcal{P}}]$.
By several applications of the coincidence lemma (which for system F is Lemma 9.13) we see that m' is already in the right set (use $\alpha \notin \mathsf{FV}(\sigma)$). Because we may assume $\alpha \notin \vec\alpha$ the substitution lemma also gives the desired property of s'. □

As for IMIT we get as corollaries that $\mathsf{SN} = \Lambda$ and that every term is strongly normalizing.

A.4 Concluding remarks

For the systems of monotone inductive types we preferred to monotonize Φ by help of monotonicity witnesses and then to replace Φ by the monotonized variant in the definition of the operators Ψ_I and Ψ_E in order to get $\Psi_I^{\supseteq}$ and $\Psi_E^{\subseteq}$[8]. In the introduction-based proof for varEMIT and in the elimination-based proof for varIMIT this approach failed and therefore a more primitive monotonization had to happen directly in the definition of $M_I^{\supseteq}(\mathcal{M})$ and $M_E^{\subseteq}(\mathcal{M})$. It is easy to see that

$$M_I^{\supseteq}(\mathcal{M}) = \bigcup\{M_I(\mathcal{M}')|\mathcal{M}' \subseteq \mathcal{M}\} \quad \text{in the proof for varEMIT}$$

and

$$M_E^{\subseteq}(\mathcal{M}) = \bigcap\{M_E(\mathcal{M}')|\mathcal{M}' \supseteq \mathcal{M}\} \quad \text{in the proof for varIMIT}$$

which is quite similar to the definitions of upper and lower monotonization. By applying the saturated closure monotone operators $\Psi_I^{\supseteq}$ and $\Psi_E^{\subseteq}$ were defined. One can show that in fact $\Psi_I^{\supseteq}$ is the upper monotonization of Ψ_I (use $M_I(\mathcal{M}) \subseteq \mathsf{SN}$) and that $\Psi_E^{\subseteq}$ is the lower monotonization of Ψ_E (use $M_E(\mathcal{M}) \cap \mathsf{SN} \in \mathsf{SAT}$). We may now ask whether the use of those canonical monotonizations would be fruitful even in case the other method works. The answer is positive. Let us consider varIMIT. We set

$$\tilde{M}_I^{\supseteq}(\mathcal{M}) = \bigcup\{M_I(\mathcal{M}')|\mathcal{M}' \subseteq \mathcal{M}\},$$

define $\tilde{\Psi}_I^{\supseteq}(\mathcal{M}) := \mathsf{cl}_{\mu\alpha\rho}(\tilde{M}_I^{\supseteq}(\mathcal{M}))$ and take for $\tilde{\mu}_I(\Phi)$ the least fixed point of $\tilde{\Psi}_I^{\supseteq}$. Then again $\tilde{\Psi}_I^{\supseteq}$ is the upper monotonization of Ψ_I and hence due to the minimality of the upper monotonization $\tilde{\Psi}_I^{\supseteq} \subseteq \Psi_I^{\supseteq}$ (pointwise). We conclude that $\tilde{\mu}_I(\Phi) \subseteq \mu_I(\Phi)$. It can be shown that $\tilde{\mu}_I(\Phi)$ is a post-fixed-point of $\Psi_E^{\subseteq}$ (Ψ_E would suffice) and hence $\tilde{\mu}_I(\Phi)$ also qualifies for the definition of the computability predicates and the normalization proof. The extracted encoding of types is

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

with $\beta \notin \{\alpha\} \cup \mathsf{FV}(\rho')$ which is exactly the encoding of types used in the embedding of UVIT into MePIT and also the encoding of types in the embedding of varIMIT into MePIT which is derived from the composition of the embeddings via MeIT and UVIT. By analysis of the proof that $\tilde{\mu}_I(\Phi)$ is a post-fixed-point of Ψ_E we even get the embedding of varIMIT into MePIT for the terms.

Now consider EMIT: If we also take the upper monotonization of Ψ_I we get another introduction-based proof for EMIT giving rise to the encoding

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

with $\beta \notin \{\alpha\} \cup \mathsf{FV}(\rho')$ which extends to an embedding of EMIT into NISPIT+ex but (trivially) not into MePIT. In section 9.5.1 we got via the encoding

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

an embedding of EMIT into NIPIT.

Unlike the situation of varIMIT we get a type encoding being free from the monotonicity statement in neither of the two approaches. Of course, we would like to get back the encoding into

[8]Clearly, the introduction-based operator has to be made bigger and the elimination-based operator has to made smaller such that in the first case a pre-fixed-point of the new operator is also a pre-fixed-point of the original operator Ψ_I and in the second case that a post-fixed-point of the new operator is also a post-fixed-point of the original operator Ψ_E.

the types of coMePIT (in chapter 6). But in the introduction-based proof for EMIT (which led to the last-mentioned encoding) we urgently needed the monotonicity assumptions in order to have that $M_I^{\supseteq}(\mathcal{M}) \supseteq M_I(\mathcal{M})$. Already the type $\forall\alpha.(\beta \to \alpha) \to \rho'$ reflects the lower monotonization which hints at the impossibility of getting rid of $\mathsf{Mon}(\lambda\alpha\rho')$ without giving up the essential $M_I^{\supseteq}(\mathcal{M}) \supseteq M_I(\mathcal{M})$. Nevertheless, it is possible even for varEMIT which does not allow the introduction-based proof by the method used in the proof for EMIT! This was shown in section 6.3.2 and section 6.3.3 by making use of coMeIT (being designed for that purpose). How is this possible? In the lattice-theoretic motivation of the embedding of varEMIT into coMeIT we used a monotone Φ and studied the rules of its lower monotonization knowing that it is the same as Φ. It turned out that for the embedding to work it was sufficient that there is a monotonicity witness. In the normalization proof however, the operator is not monotone and hence its monotonization different from it. Only via the monotonicity requirement (for EMIT this is $\forall\mathcal{P}\forall\mathcal{Q}.(\mathcal{P} \to \mathcal{Q}) \to \Phi(\mathcal{P}) \to \Phi(\mathcal{Q})$) for the monotonicity witnesses they relate to each other. Therefore we do not get a normalization proof whose extract gives the embedding of varEMIT into coMePIT. But we may take the embedding read off the first proof for EMIT, recognize that the monotonicity witnesses do not play an essential role in the embedding(!) and then optimize the embedding and finally see that it even captures varEMIT although the normalization proof we started with did not extend to varEMIT.

Appendix B

Directions for future research

In the introductory chapter the impossibility of having a "good" predecessor function on the natural numbers represented in system F was a motivation for studying extensions of system F by primitive recursion. In section 5.1.2 we saw that Gödel's system T is represented faithfully in NISPIT via the type $\mu\alpha.1+\alpha$. Therefore we have the required predecessor function. The question now is whether a general predecessor may be defined, i. e., a closed term P of type $\mu\alpha\rho \to \rho[\alpha := \mu\alpha\rho]$ such that $P(C_{\mu\alpha\rho}t) \to^* t$ for every term t of type $\rho[\alpha := \mu\alpha\rho]$. We first study the situation for monotone inductive types.
In EMIT the exact question would be: Is there a closed term P of type $\mu\alpha\rho \to \rho[\alpha := \mu\alpha\rho]$ such that $P(C_{\mu\alpha\rho}mt) \to^* t$ for every m and t? For IMIT it would be the question as above.
Let us look at the lattice-theoretic situation: If $(U, \leq)$ is a complete lattice and $\Phi : U \to U$ is a monotone, we want to show that $\mathsf{lfp}(\Phi) \leq \Phi(\mathsf{lfp}(\Phi))$. It is done by applying the rule (lfp-E^+). We have to show $\Phi(\mathsf{lfp}(\Phi) \wedge \Phi(\mathsf{lfp}(\Phi))) \leq \Phi(\mathsf{lfp}(\Phi))$. This follows by mononicity from $\mathsf{lfp}(\Phi) \wedge \Phi(\mathsf{lfp}(\Phi)) \leq \mathsf{lfp}(\Phi)$.
The proof motivates the following "definition" for EMIT:

$$P := \lambda x^{\mu\alpha\rho}.x\mathrm{E}^+_\mu\Big(m(\mu\alpha\rho \times \rho[\alpha := \mu\alpha\rho])(\mu\alpha\rho)(\lambda z^{\mu\alpha\rho\times\rho[\alpha:=\mu\alpha\rho]}.z\mathsf{L})\Big).$$

Where do we take m from? In EMIT I see no chance. Neither in IMIT, where m also has to be put after "E^+_μ". But even for a type with a fixed closed monotonicity witness there need not be *any* predecessor: Define one of the systems ESMIT by setting MTypes to the least set closed under all the type forming operations except (μ) and having $\pi := \mu\alpha.\alpha \to 1 \in \mathsf{MTypes}$. Set

$$\mathsf{map}_{\lambda\alpha.\alpha\to 1} := \Lambda\alpha\Lambda\beta\lambda f^{\alpha\to\beta}\lambda x^{\alpha\to 1}\lambda y^\beta.\mathsf{IN1}.$$

That this gives one of the systems ESMIT (and also ISMIT) is trivial to see. The constructor C_π has type $(\pi \to 1) \to \pi$. Assume that there is a term $P : \pi \to \pi \to 1$ such that $P(C_\pi t^{\pi\to 1}) \to^* t$ for every term t. Consider $\omega := \lambda x^\pi.Pxx : \pi \to 1$. Then $C_\pi\omega : \pi$. Setting $\Omega := \omega(C_\pi\omega) : 1$, we get $P(C_\pi\omega) \to^* \omega$ and hence

$$\Omega \mapsto_{\beta_\to} P(C_\pi\omega)(C_\pi\omega) \to^* \omega(C_\pi\omega) = \Omega.$$

By Lemma 2.50 we conclude that $\Omega \notin \mathsf{sn}$ contradicting strong normalization[1].
The same proof works verbatim for IMIT and by decorating C_π with $\mathsf{map}_{\lambda\alpha.\alpha\to 1}$ everywhere also for EMIT.
We observe that the monotonicity witness does not enter the specification of the predecessor. This leads to the following idea (formulated for EMIT): Define a weak predecessor to be a closed

[1]Here we need *strong* normalization.

term P of type $\mu\alpha\rho \to \rho[\alpha := \mu\alpha\rho]$ such that $P(C_{\mu\alpha\rho}mt)$ and $m(\mu\alpha\rho)(\mu\alpha\rho)(\lambda y^{\mu\alpha\rho}y)t$ have a common reduct for every m and t.
For IMIT an $m^{\forall\alpha\forall\beta.(\alpha\to\beta)\to\rho\to\rho[\alpha:=\beta]}$-predecessor is a closed term P of type $\mu\alpha\rho \to \rho[\alpha := \mu\alpha\rho]$ such that $P(C_{\mu\alpha\rho}t)$ and $m(\mu\alpha\rho)(\mu\alpha\rho)(\lambda y^{\mu\alpha\rho}y)t$ have a common reduct for every t.
For ESMIT a weak predecessor is a closed term P of type $\mu\alpha\rho \to \rho[\alpha := \mu\alpha\rho]$ such that $P(C_{\mu\alpha\rho}t)$ and $\mathsf{map}_{\lambda\alpha\rho}(\mu\alpha\rho)(\mu\alpha\rho)(\lambda y^{\mu\alpha\rho}y)t$ have a common reduct for every t.
For the example type π this would require that $P(C_\pi t)$ and $\mathsf{map}_{\lambda\alpha.\alpha\to 1}\pi\pi(\lambda y^\pi y)t$ have a common reduct. Hence, $P := \lambda x^\pi\lambda y^\pi.\mathsf{IN1}$ suffices—reflecting the triviality of the monotonicity witness.
This is the basis for the definition of systems of monotone fixed-point types. (We write now $\hat\mu$ instead of μ to emphasize the difference.) We show the elimination-based system: The (μ)-introduction rule of EMIT is kept, and the (μ)-elimination rules are replaced by:

($\hat\mu$-E) If $r^{\hat\mu\alpha\rho} \in \Lambda$, then $r\mathrm{E}_{\hat\mu} \in \Lambda$.

The rules (β_μ) and (β^+_μ) of iteration and primitive recursion are replaced by

$$\boxed{(\beta_{\hat\mu}) \qquad (C_{\hat\mu\alpha\rho}mt)\mathrm{E}_{\hat\mu} \mapsto m(\hat\mu\alpha\rho)(\hat\mu\alpha\rho)(\lambda y^{\hat\mu\alpha\rho}y)t.}$$

One can show that this extension of F is strongly normalizing either by a direct proof in the style of this thesis' normalization proofs or by an embedding into a system of non-interleaved positive fixed-point types. There are two such systems: one having the intuitive predecessor modelled by $(C_{\hat\mu\alpha\rho}t)\mathrm{E}_{\hat\mu} \mapsto t$, the other being defined as above for monotone fixed-point types but with canonical monotonicity witnesses existing due to the absence of interleaving. One may then embed them into each other (if we leave out sum types and add standard η-rules) because the canonical monotonicity witnesses then map the identity to the identity, i. e. $\mathsf{map}_{\lambda\alpha\rho}\sigma\sigma(\lambda y^\sigma y)s^{\rho[\alpha:=\sigma]} \to^* s$ for every type σ and any term s.
The extracted embeddings motivated in the sections with the direct normalization proofs embed monotone inductive types into non-interleaving positive fixed-point types, but also non-interleaving positive fixed-point types embed into non-interleaving positive inductive types (again if sum types are left out and this time because the canonical witnesses are functorial with respect to some restricted composition). We see that we may separate concerns: The behaviour of the monotonicity witnesses on the identity and their behaviour with respect to composition.
Because of these embeddings we also cannot expect an embedding of non-interleaving positive fixed-point types into IT. However, led by the proof of Tarski's theorem where it is even shown that the infimum of the pre-fixed-points is a post-fixed-point (see section 4.3.3) we can define an encoding of the system into IT[2]. The crucial definition clause is

$$(\hat\mu\text{-E}) \quad (r^{\hat\mu\alpha\rho}\mathrm{E}_{\hat\mu})' := r'\mathrm{E}_i\Big(\mathsf{map}_{\lambda\alpha\rho'}(\rho'[\alpha := (\hat\mu\alpha\rho)'])(\hat\mu\alpha\rho)'(\lambda z^{\rho[\alpha:=\hat\mu\alpha\rho]}.C_{\hat\mu\alpha\rho}z)'\Big).$$

Note that $(\lambda z^{\rho[\alpha:=\hat\mu\alpha\rho]}.C_{\hat\mu\alpha\rho}z)'$ has first[3] to be expanded according to the rule ($\hat\mu$-I) which will be the same as for the embedding of $\mathsf{varIMIT{-}rec}$ into IT but with canonical witnesses instead of any m. As expected it turns out that this encoding does not preserve reduction and hence is

[2]Unlike the situation of section 4.3.3 we cannot deal with elimination-based monotone fixed-point types because the argument for getting a post-fixed-point again uses the monotonicity of Φ which is reflected by the need of a monotonicity witness also in the rule ($\hat\mu$-E) and hence requiring to keep to systems which have selected monotonicity witnesses.

[3]Recall that in the proof of Tarski's theorem we first proved that μ_Φ is a pre-fixed-point and then used this fact for the proof that μ_Φ is also a post-fixed-point.

no embedding. As was the case for the attempt to code recursion via iteration (in section 4.5) not only functoriality is needed but also some formal induction principle. In addition one would have to define η-reduction for infimum types.
Sum types make things more complicated when trying to attack functoriality problems. One needs some η-rule and permutative conversions which may be handled though.
The question of existence of the predecessor for interleaved positive inductive types is more involved because one needs an η-rule and permutative conversions for μ-types (which are not yet known). One may at least show that in extensional equality theory (with a rule meaning that $\mu\alpha\rho$ not only expresses a weakly initial algebra but even an initial algebra, cf. [Geu92]) the intuitive definition of the predecessor shown at the beginning of this section is correct. The proofs of functoriality of map and comap which enter the argument are quite intricate and rely on naturality. For PITrec where map and comap do not use (μ-E) but (μ-E$^+$) the proofs are even more involved. Extensionally, map and comap are equal to those of $\mathsf{PITit} = \mathsf{PIT}$.
A lot more could be said but will be said or written elsewhere.
But now we come to some open problems:

- What are useful permutations of μ with μ?
- Do they strongly normalize?
- Are they confluent? (This is a problem of my current definition.)
- Do they help in analyzing the closed terms in normal form of type of a monotonicity witness, namely $\forall\alpha\forall\beta.(\alpha \to \beta) \to \rho \to \rho[\alpha := \beta]$?
- Is it at least possible to analyze the closed terms in normal form (after adding η-reduction) of type $\forall\alpha\forall\beta.(\alpha \to \beta) \to \rho \to \rho[\alpha := \beta]$ in NIPIT without sum types? Are they the canonical terms map, if one requires that they map the identity to the identity (also with η-rules included)?
- What happens with sum types? How can one reduce the set NF to have $x\mathrm{E}_+ st\vec{s} \in \mathsf{NF}$ only if $x \notin \mathsf{FV}(s) \cup \mathsf{FV}(t)$? I propose some variable elimination reduction as follows:

  $$x^{\rho+\sigma}\mathrm{E}_+ st \mapsto x\mathrm{E}_+(\lambda y^\rho.(sy)[x := \mathsf{INL}_\sigma y])(\lambda y^\sigma.(ty)[x := \mathsf{INR}_\rho y])$$

  provided that $x \in \mathsf{FV}(s) \cup \mathsf{FV}(t)$ and y is "new". It will be wise to allow this reduction also for any substitution instance. Does it preserve strong normalization? (It is correct in the extensional equality theory.)
- How about the analogous rule for μ-types?
- Is it true that for non-positive $\mu\alpha\rho$, there is no closed monotonicity witness mapping the identity to the identity? (After adding as much η-rules as possible.)
- What can be done with types such as $\mathsf{bizarre}(\rho)$ which are inhabited, monotone and not positive and do not map the identity to the identity? Are there more interesting examples of essentially monotone inductive types? Or do we first need to extend our framework to cover dependent types?[4]
- Is there a proof of confluence for a class of orthogonal term rewrite systems which includes every system of this thesis?

[4] After all, in mathematics there is a wealth of monotone functions not being positive although monotonicity is often enforced by syntactic means (cf. the example of limit inferior in the introduction).

- Is there anything to be gained by dualizing everything to coinductive types?
- Is there an embedding of SPIT into NISPIT or of NIPIT into SPIT?
- What are the differences in heights of reduction trees between PIT and NIPIT(for a term and its code in the other system)?
- Do we always have $\mu_I(\Phi) \subseteq \mu_E(\Phi)$ in the proofs of strong normalization for systems of inductive types? Are there examples which show that $\mu_I(\Phi) \neq \mu_E(\Phi)$?
- One should work out modified realizability for monotone inductive definitions using the term language of systems of monotone inductive types and prove soundness of this interpretation. (For *interleaving* positive inductive definitions without "extended induction" and without second-order universal quantification this is sketched in [Ber95].)
- Is there a meta-theorem stating that the encodings read off the normalization proofs in the style of chapter 9 are always embeddings? It seems essential that in the presented proofs all the definitions of saturated sets by comprehension are closed via the saturated closure.

Because issues of denotational semantics were never touched in this thesis I close by expressing the hope that the methods of [Loa97] may be lifted from strictly positive types to the systems of this thesis.

Index

Bibliography

[Acz77] Peter Aczel. An introduction to inductive definitions. In Barwise [Bar77], pages 739–782.

[Alt93a] Thorsten Altenkirch. *Constructions, Inductive Types and Strong Normalization*. PhD thesis, University of Edinburgh, 1993.

[Alt93b] Thorsten Altenkirch. A formalization of the strong normalization proof for system F in LEGO. In Bezem and Groote [BG93], pages 13–28.

[Alt93c] Thorsten Altenkirch. Strong normalization for T+. ASCII note, 1993.

[Bar77] Jon Barwise, editor. *Handbook of Mathematical Logic*, volume 90 of *Studies in logic and the foundations of mathematics*. North–Holland, Amsterdam, 1977.

[Bar84] Henk P. Barendregt. *The Lambda Calculus: Its Syntax and Semantics*. North–Holland, Amsterdam, second revised edition, 1984.

[Bar93] Henk P. Barendregt. Lambda calculi with types. In Samson Abramsky, Dov M. Gabbay, and Tom S. E. Maibaum, editors, *Background: Computational Structures*, volume 2 of *Handbook of Logic in Computer Science*, pages 117–309. Oxford University Press, 1993.

[BB85] Corrado Böhm and Alessandro Berarducci. Automatic synthesis of typed λ-programs on term algebras. *Theoretical Computer Science*, 39:135–154, 1985.

[BBS$^+$98] Holger Benl, Ulrich Berger, Helmut Schwichtenberg, Monika Seisenberger, and Wolfgang Zuber. Proof theory at work: Program development in the MINLOG system. In Wolfgang Bibel and Peter H. Schmitt, editors, *Automated Deduction—A Basis for Applications, Vol. II Systems and Implementation Techniques*, volume 9 of *Applied Logic Series*. Kluwer Academic Publishers, 1998.

[Ber93] Ulrich Berger. Program extraction from normalization proofs. In Bezem and Groote [BG93], pages 91–106.

[Ber95] Ulrich Berger. A constructive interpretation of positive inductive definitions. Draft, March 1995.

[BFPS81] Wilfried Buchholz, Solomon Feferman, Wolfram Pohlers, and Wilfried Sieg. *Iterated Inductive Definitions and Subsystems of Analysis: Recent Proof-Theoretical Studies*, volume 897 of *Lecture Notes in Mathematics*. Springer Verlag, 1981.

[BG93] Marc Bezem and J.F. Groote, editors. *Typed Lambda Calculi and Applications*, volume 664 of *Lecture Notes in Computer Science*. Springer Verlag, 1993.

[BTG91] Val Breazu-Tannen and Jean Gallier. Polymorphic rewriting preserves algebraic strong normalization. *Theoretical Computer Science*, 83(1):3–28, 1991.

[CV97] Venanzio Capretta and Silvio Valentini. A general method to prove the normalization theorem for first and second order typed λ-calculi. To be published in *Mathematical Structures in Computer Science*, 1997.

[Dav55] Anne C. Davis. A characterization of complete lattices. *Pacific Journal of Mathematics*, 5:311–319, 1955.

[dB72] N. G. de Bruijn. Lambda calculus notation with nameless dummies, a tool for automatic formula manipulation, with application to the Church-Rosser theorem. *Indagationes Mathematicae*, 34:381–392, 1972.

[Fef77] Solomon Feferman. Theories of finite type related to mathematical practice. In Barwise [Bar77], pages 913–971.

[Fef82] Solomon Feferman. Monotone inductive definitions. In Anne S. Troelstra and Dirk van Dalen, editors, *The L. E. J. Brouwer Centenary Symposium*, pages 77–89. North–Holland, Amsterdam, 1982.

[Gal90] Jean H. Gallier. On Girard's "candidats de reductibilité". In Odifreddi [Odi90], pages 123–203.

[Gal98] Jean Gallier. Typing untyped λ-terms, or reducibility strikes again! *Annals of Pure and Applied Logic*, 91:231–270, 1998.

[Geu92] Herman Geuvers. Inductive and coinductive types with iteration and recursion. In Bengt Nordström, Kent Pettersson, and Gordon Plotkin, editors, *Proceedings of the 1992 Workshop on Types for Proofs and Programs, Båstad, Sweden, June 1992*, pages 193–217, 1992. Electronically available via `ftp://ftp.cs.chalmers.se/pub/cs-reports/baastad.92/proc.dvi.Z`.

[Geu93] Herman Geuvers. *Logics and Type Systems*. Proefschrift (PhD thesis), University of Nijmegen, September 1993.

[Gir72] Jean-Yves Girard. *Interprétation fonctionnelle et élimination des coupures dans l'arithmétique d'ordre supérieur*. Thèse de Doctorat d'État, Université de Paris VII, 1972.

[GLT89] Jean-Yves Girard, Yves Lafont, and Paul Taylor. *Proofs and Types*, volume 7 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 1989.

[Gog94] Healfdene Goguen. *A Typed Operational Semantics for Type Theory*. PhD thesis, University of Edinburgh, August 1994. Available as LFCS Report ECS-LFCS-94-304.

[Gog95] Healfdene Goguen. Typed operational semantics. In Mariangiola Dezani-Ciancaglini and Gordon Plotkin, editors, *Proceedings of the Second International Conference on Typed Lambda Calculi and Applications (TLCA '95), Edinburgh, United Kingdom, April 1995*, volume 902 of *Lecture Notes in Computer Science*, pages 186–200. Springer Verlag, 1995.

[GRS97] Thomas Glass, Michael Rathjen, and Andreas Schlüter. On the proof-theoretic strength of monotone induction in explicit mathematics. *Annals of Pure and Applied Logic*, 85:1–46, 1997.

[How80] W. A. Howard. The formulae-as-types notion of construction. In J. P. Seldin and J. R. Hindley, editors, *To H. B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*, pages 479–490. Academic Press, 1980.

[How92] Brian Howard. *Fixed Points and Extensionality in Typed Functional Programming Languages*. PhD thesis, Stanford University, 1992.

[JO97] Jean-Pierre Jouannaud and Mitsuhiro Okada. Abstract data type systems. *Theoretical Computer Science*, 173:349–391, 1997.

[Joa97] Felix Joachimski. On η-expansion in Gödel's $\mathcal{T}$, pure type systems and calculi with permutative conversions. Unpublished manuscript, September 1997.

[Kol85] George Koletsos. Church-Rosser theorem for typed functional systems. *The Journal of Symbolic Logic*, 50(3):782–790, 1985.

[Lei75] Daniel Leivant. Strong normalization for arithmetic (variations on a theme of Prawitz). In Justus Diller and G. H. Müller, editors, *Proof Theory Symposion, Kiel, Germany, 1974*, volume 500 of *Lecture Notes in Mathematics*, pages 182–197. Springer Verlag, 1975.

[Lei90] Daniel Leivant. Contracting proofs to programs. In Odifreddi [Odi90], pages 279–327.

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

[Loa97] Ralph Loader. Equational theories for inductive types. *Annals of Pure and Applied Logic*, 84:175–217, 1997.

[Men87a] 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, Ithaca, N.Y.*, pages 30–36. IEEE Computer Society Press, 1987. Forms a part of [Men87b].

[Men87b] Paul F. Mendler. Inductive definition in type theory. Technical Report 87-870, Cornell University, Ithaca, N.Y., September 1987. PhD Thesis (Paul F. Mendler = Nax P. Mendler).

[Mit96] John C. Mitchell. *Foundations for Programming Languages*. Foundations of Computing. The MIT Press, 1996.

[MKO95] David McAllester, Jakov Kučan, and Daniel Otth. A proof of strong normalization for F_2, F_ω, and beyond. *Information and Computation*, 121(2):193–200, 1995.

[ML84] Per Martin-Löf. *Intuitionistic Type Theory*. Bibliopolis, Napoli, 1984.

[MN98] Richard Mayr and Tobias Nipkow. Higher-order rewrite systems and their confluence. *Theoretical Computer Science*, 192:3–29, 1998.

[Mos74] Yiannis N. Moschovakis. *Elementary Induction on Abstract Structures*, volume 77 of *Studies in Logic and the Foundations of Mathematics*. North–Holland, Amsterdam, 1974.

[Odi90] Piergiorgio Odifreddi, editor. *Logic and Computer Science*, volume 31 of *APIC Studies in Data Processing*. Academic Press, 1990.

[Par92] Michel Parigot. Recursive programming with proofs. *Theoretical Computer Science*, 94:335–356, 1992.

[Pra71] Dag Prawitz. Ideas and results in proof theory. In Jens E. Fenstad, editor, *Proceedings of the Second Scandianvian Logic Symposium*, volume 63 of *Studies in Logic and the Foundations of Mathematics*, pages 235–307. North–Holland, Amsterdam, 1971.

[Rat96] Michael Rathjen. Monotone inductive definitions in explicit mathematics. *The Journal of Symbolic Logic*, 61:125–146, 1996.

[Rat98] Michael Rathjen. Explicit mathematics with the monotone fixed point principle. *The Journal of Symbolic Logic*, 63:181–200, 1998.

[Rat99] Michael Rathjen. Explicit mathematics with the monotone fixed point principle. II: Models. *The Journal of Symbolic Logic*, 1999? To appear.

[Rei96] Gunnar Reitel. Eine Realisierbarkeitsinterpretation der Arithmetik zweiter Stufe. Diplomarbeit (Master's thesis), Ludwig-Maximilians-Universität München, September 1996. In German.

[Rey74] John C. Reynolds. Towards a theory of type structure. In B. Robinet, editor, *Programming Symposium*, volume 19 of *Lecture Notes in Computer Science*, pages 408–425, Berlin, 1974. Springer-Verlag.

[Sce89] Andre Scedrov. Normalization revisited. *Contemporary Mathematics*, 92:357–369, 1989.

[Str99] Thomas Strahm. First steps into metapredicativity in explicit mathematics. In S. Barry Cooper and John K. Truss, editors, *Sets and Proofs. Invited Papers from the Logic Colloquium 97, University of Leeds, England, July 6–13, 1997*. Cambridge University Press, 1999? To appear.

[Tai67] William W. Tait. Intensional interpretations of functionals of finite type I. *The Journal of Symbolic Logic*, 32(2):198–212, 1967.

[Tai75] William W. Tait. A realizability interpretation of the theory of species. In R. Parikh, editor, *Logic Colloquium Boston 1971/72*, volume 453 of *Lecture Notes in Mathematics*, pages 240–251. Springer Verlag, 1975.

[Tak95] Masako Takahashi. Parallel reduction in λ-calculus. *Information and Computation*, 118(1):120–127, 1995.

[Tar55] Alfred Tarski. A lattice-theoretical fixpoint theorem and its applications. *Pacific Journal of Mathematics*, 5:285–309, 1955.

[Tat94] Makoto Tatsuta. Two realizability interpretations of monotone inductive definitions. *International Journal of Foundations of Computer Science*, 5(1):1–21, 1994.

[UV97] Tarmo Uustalu and Varmo Vene. A cube of proof systems for the intuitionistic predicate μ-, ν-logic. In Magne Haveraaen and Olaf 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*, pages 237–246, May 1997.

[vdP96a] Jaco van de Pol. Termination of higher-order rewrite systems. Quaestiones Infinitae XVI, Department of Philosophy, Utrecht University, 1996. Proefschrift (PhD thesis).

[vdP96b] Jaco van de Pol. Two *different* strong normalization proofs? In G. Dowek, J. Heering, K. Meinke, and B. Möller, editors, *Proceedings of the Second International Workshop on Higher-Order Algebra, Logic and Term Rewriting (HOA '95), Paderborn, Germany*, volume 1074 of *Lecture Notes in Computer Science*, pages 201–220. Springer Verlag, 1996. Forms a part of [vdP96a].

[vR93] Femke van Raamsdonk. Confluence and superdevelopments. In Claude Kirchner, editor, *Proceedings of the 5th International Conference on Rewriting Techniques and Applications (RTA '93), Montreal, Canada, June 1993*, volume 690 of *Lecture Notes in Computer Science*, pages 168–182. Springer Verlag, 1993.

[vR96] Femke van Raamsdonk. *Confluence and Normalisation for Higher-Order Rewriting.* Academisch Proefschrift (PhD thesis), Vrije Universiteit te Amsterdam, 1996.

[vRS95] Femke van Raamsdonk and Paula Severi. On normalisation. Technical Report CS-R9545, Centrum voor Wiskunde en Informatica (CWI), Amsterdam, June 1995.