Program extraction from normalization proofs

Helmut Schwichtenberg
(with Ulrich Berger, Stefan Berghofer and Pierre Letouzey)

Mathematisches Institut, Universität München

Mathematics - Algorithms - Proofs, 9. January 2006

Goal

Formally extract computational content of an interesting proof: existence of normal forms in simply typed λ-calculus.

Related work: (1) Normalization algorithms have been machine-extracted from formal proofs in the type-theoretic proof checker ALF (the precursor of AGDA) (C. Coquand 1994, T. Coquand and Dybjer 1997).
However, there the main ingredients of "normalization by evaluation", the evaluation function and its inverse $\downarrow$, show up explicitely in the proofs already. Here: these components of the algorithm are implicit in the logical argument and are made explicit by the extraction only.

Goal

Formally extract computational content of an interesting proof: existence of normal forms in simply typed λ-calculus.

Related work: (1) Normalization algorithms have been machine-extracted from formal proofs in the type-theoretic proof checker ALF (the precursor of AGDA) (C. Coquand 1994, T. Coquand and Dybjer 1997).
However, there the main ingredients of "normalization by evaluation", the evaluation function and its inverse $\downarrow$, show up explicitely in the proofs already. Here: these components of the algorithm are implicit in the logical argument and are made explicit by the extraction only.

Related work (ctd.)

(2) There exist also formalized normalization proofs for systems such as

- the Calculus of Constructions (Altenkirch 1994),
- System F (Altenkirch, Hofmann and Streicher 1996),
- the typed λ-calculus with co-products (Altenkirch, Dybjer, Hofmann, Scott 2001) and
- λ-calculi with various weak reduction strategies (Danvy 2005).

However, no program extraction by machine has been carried out.

β-reduction, η-expansion

(Simple) type: $\iota \mid \rho \Rightarrow \sigma$ (may also include $\rho \times \sigma$).
(Simply typed λ-) term: $x^\sigma \mid (\lambda x^\rho r^\sigma)^{\rho\Rightarrow\sigma} \mid (r^{\rho\Rightarrow\sigma} s^\rho)^\sigma$.
Λ_ρ denotes the set of all terms of type ρ.

$$\begin{aligned} (\lambda x r)s &\mapsto r_x[s] && \beta\text{-conversion}, \\ r &\mapsto \lambda x.\, rx && \eta\text{-expansion } (x \notin \mathrm{FV}(r)). \end{aligned}$$

Definition

r is in β-normal form if no (inner) β-conversion is possible.

Definition

Let r be in β-normal form. r is in η-long normal form if no (inner) η-expansion is possible without creating a new β-convertible subterm.

Long normal forms

Terms in long normal form (i.e. normal w.r.t. β-reduction and η-expansion) are inductively defined by

$$\lambda x r \mid (x r_1 \dots r_n)^\iota.$$

Definition

Let r be in β-normal form. With $\mathrm{lnf}(r)$ we denote the result of maximally η-expanding r.

Example

Let $x, y : \iota$ and $f : \iota \to \iota$ and $G : (\iota \to \iota) \to \iota \to \iota$.

$$f \to_\eta \lambda x.fx,$$
$$G \to_\eta \lambda f.Gf \to_\eta \lambda f.G(\lambda x.fx) \to_\eta \lambda f \lambda y.G(\lambda x.fx)y = \mathrm{lnf}(G).$$

Long normal forms

Terms in long normal form (i.e. normal w.r.t. β-reduction and η-expansion) are inductively defined by

$$\lambda x r \mid (x r_1 \dots r_n)^\iota.$$

Definition

Let r be in β-normal form. With $\mathrm{lnf}(r)$ we denote the result of maximally η-expanding r.

Example

Let $x, y : \iota$ and $f : \iota \to \iota$ and $G : (\iota \to \iota) \to \iota \to \iota$.

$$f \to_\eta \lambda x.fx,$$
$$G \to_\eta \lambda f.Gf \to_\eta \lambda f.G(\lambda x.fx) \to_\eta \lambda f \lambda y.G(\lambda x.fx)y = \mathrm{lnf}(G).$$

Term families

Problem: How to represent bound variables in terms?

Solution: "term families", that is parametrized versions of terms.

Main idea: the term family of M at index k reproduces M with bound variables renamed starting at k.

Example

For $M := \lambda u \lambda v.c(\lambda x.vx)(\lambda y \lambda z.zu)$ the associated term family M^∞ at index 3 yields

$$M^\infty(3) := \lambda x_3 \lambda x_4.c(\lambda x_5.x_4 x_5)(\lambda x_5 \lambda x_6.x_6 x_3).$$

Notation for terms: $M, N, K, \ldots$; for term families: $r, s, t, \ldots$.

Term families

Problem: How to represent bound variables in terms?

Solution: "term families", that is parametrized versions of terms.

Main idea: the term family of M at index k reproduces M with bound variables renamed starting at k.

Example

For $M := \lambda u \lambda v.c(\lambda x.vx)(\lambda y \lambda z.zu)$ the associated term family M^∞ at index 3 yields

$$M^\infty(3) := \lambda x_3 \lambda x_4.c(\lambda x_5.x_4 x_5)(\lambda x_5 \lambda x_6.x_6 x_3).$$

Notation for terms: $M, N, K, \ldots$; for term families: $r, s, t, \ldots$.

Term families (ctd.)

Definition

To every term M^ρ assign a term family $M^\infty : \mathbb{N} \to \Lambda_\rho$:

$$\begin{aligned} x^\infty(k) &:= x, \\ (\lambda y M)^\infty(k) &:= \lambda x_k (M_y[x_k]^\infty(k+1)), \\ (MN)^\infty(k) &:= M^\infty(k) N^\infty(k). \end{aligned}$$

Application of $r : \mathbb{N} \to \Lambda_{\rho \Rightarrow \sigma}$ to $s : \mathbb{N} \to \Lambda_\rho$ is $rs : \mathbb{N} \to \Lambda_\sigma$

$$(rs)(k) := r(k)s(k).$$

Let $k > \mathrm{FV}(M)$ mean that k is greater than all i such that $x_i^\rho \in \mathrm{FV}(M)$ for some type ρ.

If $M =_\alpha N$, then $M^\infty = N^\infty$

Proof.

Induction on $\mathrm{ht}(M)$. Only the case where M and N are abstractions is critical. So assume $\lambda y^\rho M =_\alpha \lambda z^\rho N$. Then $M_y[P] =_\alpha N_z[P]$ for all terms P of type ρ. In particular $M_y[x_k] =_\alpha N_z[x_k]$ for arbitrary $k \in \mathbb{N}$. Therefore

$$\begin{aligned}(\lambda y M)^\infty(k) &= \lambda x_k(M_y[x_k]^\infty(k+1)) \\ &= \lambda x_k(N_z[x_k]^\infty(k+1)) \quad \text{by IH} \\ &= (\lambda z N)^\infty(k).\end{aligned}$$

□

If $k > \mathrm{FV}(M)$, then $M^\infty(k) =_\alpha M$

Proof.

Induction on $\mathrm{ht}(M)$. We only consider the case $\lambda y M$. The assumption $k > \mathrm{FV}(\lambda y M)$ implies $x_k \notin \mathrm{FV}(\lambda y M)$ and hence $\lambda y M =_\alpha \lambda x_k(M_y[x_k])$.

$$\begin{aligned}(\lambda y M)^\infty(k) &= \lambda x_k(M_y[x_k]^\infty(k+1)) \\ &=_\alpha \lambda x_k(M_y[x_k]) \quad \text{by IH, for } k{+}1 > \mathrm{FV}(M_y[x_k]) \\ &=_\alpha \lambda y M.\end{aligned}$$

□

Extraction from term families

Let $\mathrm{ext}(r) := r(k)$, where k is the least number greater than all i such that some variable of the form x_i^ρ occurs (free or bound) in $r(0)$.

Lemma

$\mathrm{ext}\big(M^\infty\big) =_\alpha M$.

Proof.

$\mathrm{ext}\big(M^\infty\big) = M^\infty(k)$ for the least $k > i$ for all i such that x_i^ρ occurs (free or bound) in $M^\infty(0)$, hence $k > \mathrm{FV}(M)$. Now use the last lemma. □

Reify and reflect

$[\![\iota]\!] := \Lambda_\iota^{\mathbb{N}}$, $[\![\rho \Rightarrow \sigma]\!] := [\![\sigma]\!]^{[\![\rho]\!]}$ (full function spaces). Define

$$\downarrow_\rho : [\![\rho]\!] \to (\mathbb{N} \to \Lambda_\rho) \ (\text{“reify”}) \quad \uparrow_\rho : (\mathbb{N} \to \Lambda_\rho) \to [\![\rho]\!] \ (\text{“reflect”}),$$

simultaneously, by induction on ρ:

$$\downarrow_\iota(r) := r, \qquad \uparrow_\iota(r) := r,$$
$$\downarrow_{\rho\Rightarrow\sigma}(a)(k) := \lambda x_k^\rho.\downarrow_\sigma\big(a(\uparrow_\rho(x_k^\infty))\big)(k+1), \quad \uparrow_{\rho\Rightarrow\sigma}(r)(b) := \uparrow_\sigma(r\downarrow_\rho(b)).$$

Then, for $a_i \in [\![\rho_i]\!]$

$$\uparrow_{\vec{\rho}\Rightarrow\sigma}(r)(a_1, \ldots a_n) = \uparrow_\sigma(r\downarrow_{\rho_1}(a_1)\ldots\downarrow_{\rho_n}(a_n)) \qquad (1)$$

Correctness of nbe

Lemma

$\downarrow([\![M]\!]_\uparrow) = M^\infty$ *for* M *in long normal form.*

Proof.

Induction on the height of M. **Case** $\lambda y^\rho M^\sigma$.

$$\begin{aligned}
\downarrow([\![\lambda y M]\!]_\uparrow)(k) &= \lambda x_k\big(\downarrow([\![\lambda y M]\!]_\uparrow(\uparrow(x_k^\infty)))(k+1)\big) \\
&= \lambda x_k\big(\downarrow([\![M_y[x_k]]\!]_\uparrow)(k+1)\big) \\
&= \lambda x_k\big(M_y[x_k]^\infty(k+1)\big) && \text{by IH} \\
&= (\lambda y M)^\infty(k).
\end{aligned}$$

Case $(x\vec{M})^\iota$. By (1) and IH,
$[\![x\vec{M}]\!]_\uparrow = \uparrow(x^\infty)([\![\vec{M}]\!]_\uparrow) = x^\infty\downarrow([\![\vec{M}]\!]_\uparrow) = x^\infty\vec{M}^\infty = \big(x\vec{M}\big)^\infty$. □

Domain semantics

We refine the naive model of "full" function spaces by spaces of "partial continuous functionals".

That is, we take the more approximative view of domain theory towards higher type objects, as is appropriate for our present task of program extraction.

We sketch one particular approach to domain theory: Scott's theory of information systems (1982).

Basic idea of information systems

Provide an axiomatic setting for describing approximations of abstract objects (like functions or functionals) by concrete, finite ones.

We take an arbitrary countable set A of "bits of data" or "tokens" as a basic notion to be explained axiomatically. In order to use such data for building approximations of abstract objects, we need a notion of "consistency" (for finite sets of tokens).

We also need an "entailment relation" between consistent sets X of data and single tokens a ("the information in X is sufficient to compute the bit of information a").

Definition of information systems

An information system is a structure $(A, \mathrm{Con}, \vdash)$ where A is a countable set (the tokens), Con is a nonempty set of finite subsets of A (the consistent sets), and $\vdash$ is a subset of $\mathrm{Con} \times A$ (the entailment relation) which satisfy

(a) If $X \subseteq Y \in \mathrm{Con}$ then $X \in \mathrm{Con}$;

(b) If $a \in A$ then $\{a\} \in \mathrm{Con}$;

(c) If $X \vdash a$ then $X \cup \{a\} \in \mathrm{Con}$;

(d) If $X \in \mathrm{Con}$ and $a \in X$ then $X \vdash a$;

(e) If $X, Y \in \mathrm{Con}$, $X \vdash Y$ and $Y \vdash c$, then $X \vdash c$.

$X \vdash Y$ abbreviates $\forall b{\in}Y(X \vdash b)$.

Objects of an information system

The objects or ideals of an information system $\mathbf{A} = (A, \mathrm{Con}, \vdash)$ are defined to be those subsets z of A which satisfy

(a) $X \subseteq^{\mathrm{fin}} z$ implies $X \in \mathrm{Con}$ (z is "consistent");

(b) $X \subseteq^{\mathrm{fin}} z$ and $X \vdash a$ implies $a \in z$ (z is "deductively closed").

The set of all objects of $\mathbf{A}$ is written $|\mathbf{A}|$.

Information systems and Scott domains

The structure $(|\mathbf{A}|, \subseteq, \emptyset)$ is a Scott domain (countably based algebraic dcpo), whose set of compact elements can be represented as $|\mathbf{A}|_c = \{\, \overline{X} \mid X \in \mathrm{Con} \,\}$ where $\overline{X} := \{\, a \in A \mid X \vdash a \,\}$.

The converse is true as well: every domain with countable basis can be represented as the set of objects of an information system.

Effective information systems

An information system $\mathbf{A} = (A, \mathrm{Con}, \vdash)$ is effective if the sets A and Con and the relation $\vdash$ are decidable (w.r.t. a given coding). An object of $\mathbf{A}$, that is, an element of $|\mathbf{A}|$ is called computable if it is recursively enumerable.

All information systems and constructions of information systems will be effective in the sense that they turn effective information systems into effective information systems in an effective way, that is, from Gödel number of the input systems a Gödel number of the output system can be computed.

Function spaces

Let $\mathbf{A}$ and $\mathbf{B}$ be information systems. Then the objects u of $\mathbf{A} \to \mathbf{B}$ are in bijective correspondence with the continuous functions f from $|\mathbf{A}|$ to $|\mathbf{B}|$:

- For $u\colon \mathbf{A} \to \mathbf{B}$ define $|u|\colon |\mathbf{A}| \to |\mathbf{B}|$ by

$$|u|(z) := \{\, b \in B \mid u(X, b) \text{ for some } X \subseteq^{\mathrm{fin}} z \,\}.$$

- For $f\colon |\mathbf{A}| \to |\mathbf{B}|$ define $\hat{f}\colon \mathbf{A} \to \mathbf{B}$ by

$$\hat{f}(X, b) := b \in f(\overline{X}).$$

Moreover, $f = |\hat{f}|$ and $u = \widehat{|u|}$.

A universal information system

$$\mathbf{C}_\iota := \mathbf{N} \to \mathbf{\Lambda} \qquad \mathbf{C}_\omega := \sum_\rho \mathbf{C}_\rho$$
$$\mathbf{C}_{\rho\Rightarrow\sigma} := \mathbf{C}_\rho \to \mathbf{C}_\sigma$$

Have injections and projections

$$\mathrm{in}_\rho : |\mathbf{C}_\rho| \to |\mathbf{C}_\omega|, \qquad \mathrm{in}_\rho^{-1} : |\mathbf{C}_\omega| \to |\mathbf{C}_\rho|.$$

Every $a \in |\mathbf{C}_\omega|$ has the form $\mathrm{in}_\rho(u)$ for some $u \in |\mathbf{C}_\rho|$, or else is $\overline{\emptyset}$.

Call $\{\, \mathrm{in}_\rho(u) \mid u \in |\mathbf{C}_\rho| \,\}$ the ρ-part of $|\mathbf{C}_\omega|$.

Let $P_\rho(a)$ mean that a is in the ρ-part of $|\mathbf{C}_\omega|$.

Administrative functions: ModIota, HatIota

Define $\mathrm{ModIota}\colon |\mathbf{C}_\omega| \to |\mathbf{N}| \to |\mathbf{\Lambda}|$ by

$$\mathrm{ModIota}\big(\mathrm{in}_\iota(u)\big) := |u|$$

$\mathrm{ModIota}\big(\mathrm{in}_\tau(u)\big)(k) := \overline{\emptyset}$ for $\tau \neq \iota$, $\mathrm{ModIota}\big(\overline{\emptyset}\big)(k) := \overline{\emptyset}$.

For $g\colon |\mathbf{N}| \to |\mathbf{\Lambda}|$ we have $\hat{g} \in |\mathbf{N} \to \mathbf{\Lambda}| = |\mathbf{C}_\iota|$, hence $\mathrm{in}_\iota(\hat{g}) \in |\mathbf{C}_\omega|$. Define $\mathrm{HatIota}\colon \big(|\mathbf{N}| \to |\mathbf{\Lambda}|\big) \to |\mathbf{C}_\omega|$ by

$$\mathrm{HatIota}(g) := \mathrm{in}_\iota(\hat{g}).$$

for $g\colon |\mathbf{N}| \to |\mathbf{\Lambda}|$, and $:= \overline{\emptyset}$ else. Then

$$\mathrm{ModIota}(\mathrm{HatIota}(g)) = g.$$

Administrative functions: Mod, Hat

Define $\mathrm{Mod}\colon |\mathbf{C}_\omega| \to |\mathbf{C}_\omega| \to |\mathbf{C}_\omega|$ by

$$\mathrm{Mod}\big(\mathrm{in}_{\rho\Rightarrow\sigma}(u)\big) := \mathrm{in}_\sigma \circ |u| \circ \mathrm{in}_\rho^{-1},$$

$\mathrm{Mod}\big(\mathrm{in}_\tau(u)\big)(a) := \overline{\emptyset}$ for τ not of arrow form, and
$\mathrm{Mod}\big(\overline{\emptyset}\big)(a) := \overline{\emptyset}$.

Define $\mathrm{Hat}_{\rho,\sigma}\colon \big(|\mathbf{C}_\omega| \to |\mathbf{C}_\omega|\big) \to |\mathbf{C}_\omega|$ by

$$\mathrm{Hat}_{\rho,\sigma}(h) := \mathrm{in}_{\rho\Rightarrow\sigma}(\hat{f})$$

for $f := \mathrm{in}_\sigma^{-1} \circ h \circ \mathrm{in}_\rho\colon |\mathbf{C}_\rho| \to |\mathbf{C}_\sigma|$. Then

$$\mathrm{Mod}(\mathrm{Hat}_{\rho,\sigma}(h)) = \mathrm{in}_\sigma \circ \mathrm{in}_\sigma^{-1} \circ h \circ \mathrm{in}_\rho \circ \mathrm{in}_\rho^{-1}.$$

Evaluation and currying

Lemma

Let $\mathbf{A}$ *and* $\mathbf{B}$ *be information systems. Then*

$$\mathrm{eval}\colon (\mathbf{A} \to \mathbf{B}) \times \mathbf{A} \to \mathbf{B}$$
$$\mathrm{eval}(f, x) := f(x) \quad (f \in |\mathbf{A} \to \mathbf{B}|,\ x \in |\mathbf{A}|)$$

is continuous.

Lemma

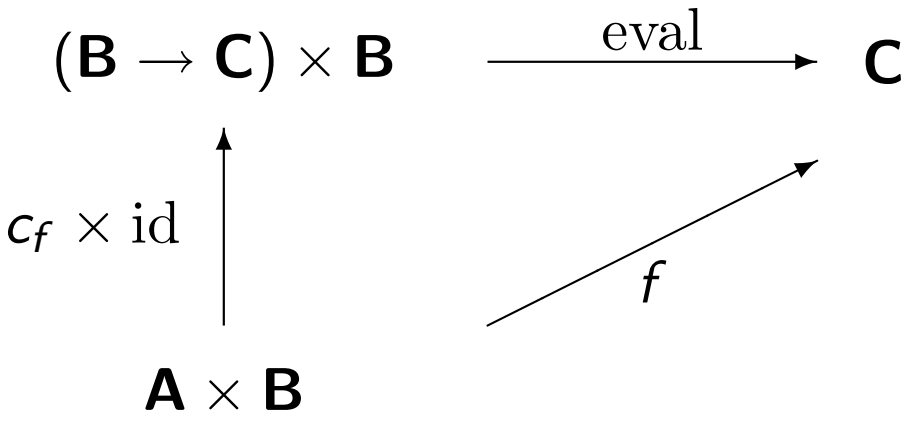
Let $\mathbf{A}$, $\mathbf{B}$ *and* $\mathbf{C}$ *be information systems with* $A \cap B = \emptyset$. *Then*

$$\mathrm{curry}\colon (\mathbf{A} \times \mathbf{B} \to \mathbf{C}) \to (\mathbf{A} \to (\mathbf{B} \to \mathbf{C}))$$
$$\mathrm{curry}(f)(x)(y) := f(x, y) \quad (f \in |\mathbf{A} \times \mathbf{B} \to \mathbf{C}|,\ x \in |\mathbf{A}|,\ y \in |\mathbf{B}|)$$

is well-defined and continuous.

Information systems as a cartesian closed category

Hence: for any continuous $f : \mathbf{A} \times \mathbf{B} \to \mathbf{C}$ there is a continuous $c_f : \mathbf{A} \to (\mathbf{B} \to \mathbf{C})$ such that $\mathrm{eval} \circ (c_f \times \mathrm{id}) = f$:



"Terminal" information system: $(\emptyset, \{\emptyset\}, \emptyset)$. Hence: information systems with continuous maps form a cartesian closed category. The importance of this category, from our computational point of view, is that it also allows least fixed points.

Meaning of typed λ-terms

An important consequence of working in a cartesian closed category is that every "explicit definition" involving higher types actually defines an object.

Lemma

For every λ-term r^ρ and every list $x_1^{\rho_1}, \ldots, x_n^{\rho_n}$ of variables containing all variables free in r we have $[\![r]\!] : \mathbf{C}_{\rho_1} \times \cdots \times \mathbf{C}_{\rho_n} \to \mathbf{C}_\rho$ such that for all $u_1 \in |\mathbf{C}_{\rho_1}|, \ldots, u_n \in |\mathbf{C}_{\rho_n}|$

$$
\begin{aligned}
[\![x_i]\!](u_1, \ldots, u_n) &= u_i \\
[\![rs]\!](u_1, \ldots, u_n) &= [\![r]\!](u_1, \ldots, u_n)([\![s]\!](u_1, \ldots, u_n)) \\
[\![\lambda x r]\!](u_1, \ldots, u_n)(v) &= [\![r]\!](u_1, \ldots, u_n, v) \\
[\![\langle r, s \rangle]\!](u_1, \ldots, u_n) &= ([\![r]\!](u_1, \ldots, u_n), [\![s]\!](u_1, \ldots, u_n)) \\
[\![ri]\!](u_1, \ldots, u_n) &= \pi_i([\![r]\!](u_1, \ldots, u_n)) \quad (i = 0, 1).
\end{aligned}
$$

Proof

By induction on r. In case r is a variable x_i, let $[\![x_i]\!] := \pi_i^n$, where π_i^n is the i-th projection. For application, define $[\![rs]\!] := \mathrm{eval} \circ ([\![r]\!] \times [\![s]\!])$. This is continuous, and

$$\begin{aligned}
[\![rs]\!](u_1, \ldots, u_n) &= \mathrm{eval}\big(([\![r]\!] \times [\![s]\!])(u_1, \ldots, u_n)\big) \\
&= \mathrm{eval}([\![r]\!](u_1, \ldots, u_n), [\![s]\!](u_1, \ldots, u_n)) \\
&= [\![r]\!](u_1, \ldots, u_n)([\![s]\!](u_1, \ldots, u_n)).
\end{aligned}$$

For a λ-abstraction, let $[\![\lambda x r]\!] := \mathrm{curry}([\![r]\!])$. Then $[\![\lambda x r]\!](u_1, \ldots, u_n)(v) = \mathrm{curry}([\![r]\!])(u_1, \ldots, u_n)(v) = [\![r]\!](u_1, \ldots, u_n, v)$. For pairing and projection the argument is similar.

Predicates

$$
\begin{aligned}
N(r,s) &:= r \to \cdots \to t \text{ for some } \beta\text{-normal } t, \text{ and } \mathrm{Inf}(t){=}s \\
A(r,s) &:= r = x r_1 \ldots r_n \text{ and } s = x s_1 \ldots s_n, \text{ with } N(r_i, s_i) \\
H(r,s) &:= r = (\lambda x.t) u \vec{t} \text{ and } s = t_x[u]\vec{t} \\
F(r,k) &:= \text{every index of a variable free in } r \text{ is } < k
\end{aligned}
$$

Let $\mathrm{FN}(r) := \forall k.F(r,k) \to \exists s\, N(r,s)$, similarly $\mathrm{FA}(r)$.

A property of the N-predicate

$N(r, s) :=$ for some term t, $r \longrightarrow \cdots \longrightarrow t$ β-normal, and $\mathrm{lnf}(t) = s$.

Lemma

If $N(rx, s)$ with $x \notin \mathrm{FV}(r)$, then $N(r, \lambda x^{\rho} s)$.

Proof.

Assume $N(rx, s)$, i.e., $rx \longrightarrow \cdots \longrightarrow t$ β-normal, and $\mathrm{lnf}(t) = s$.
Show $N(r, \lambda x^{\rho} s)$. Consider $rx \longrightarrow \cdots \longrightarrow t$.
Case $rx \longrightarrow \cdots \longrightarrow r_n x \longrightarrow t_{n+1} \longrightarrow \cdots \longrightarrow t$, $t_{n+1} \neq r_{n+1} x$. Then $r_n = \lambda x r'_n$, $t_{n+1} = r'_n$. Hence $r \longrightarrow \cdots \longrightarrow r_n = \lambda x t_{n+1} \longrightarrow \cdots \longrightarrow \lambda x t$ β-normal, and $\mathrm{lnf}(\lambda x t) = \lambda x \mathrm{lnf}(t) = \lambda x s$.
Case $rx \longrightarrow \cdots \longrightarrow r_n x = t$ β-normal, and $\mathrm{lnf}(t) = s$. Then $r \longrightarrow \cdots \longrightarrow r_n$ β-normal, and $\mathrm{lnf}(r_n) = \lambda x \mathrm{lnf}(r_n x) = \lambda x s$. □

Computability predicates

$$\begin{aligned} C^\iota(r) &:= \mathrm{FN}^\iota(r), \\ C^{\rho\Rightarrow\sigma}(r) &:= \forall^{\mathrm{nc}} s.C^\rho(s) \to C^\sigma(rs). \end{aligned}$$

$C^\rho(r)$ must have computational content (because FN has). We will later make it explicit.

We freely use properties of N, A, H, F as axioms, provided they have no computational content.

Example

(Ax1): $F(r,k) \to N^\sigma(rx_k,s) \to N^{\rho\Rightarrow\sigma}(r,\lambda x_k^\rho s)$.

Lemma 1: (a) $C^\rho(r) \to \mathrm{FN}^\rho(r)$, (b) $\mathrm{FA}^\rho(r) \to C^\rho(r)$

Induction on ρ. **Case** ι. (a) holds by definition. (b). Assume $\mathrm{FA}(r)$, that is $\forall k.F(r,k) \to \exists s\, A^\iota(r,s)$. We must show $C^\iota(r)$, that is $\forall k.F(r,k) \to \exists s\, N^\iota(r,s)$. Use (Ax2): $A^\iota(r,s) \to N^\iota(r,s)$.

Case $\rho \Rightarrow \sigma$. (a). Assume $C^{\rho\Rightarrow\sigma}(r)$ and $F(r,k)$. Show $\exists s\, N(r,s)$. Let $\vec{\rho} \vdash r : \rho$, $\vec{\sigma} := e(\vec{\rho}, k, \rho)$, $l := \mathrm{Lh}(\vec{\rho}, \vec{\sigma})$. Have $\mathrm{FA}^\rho(x_k)$ by $A^\rho(x_k, x_k)$ (axiom on A), hence $C^\rho(x_k)$ by IH(b). From $C^{\rho\Rightarrow\sigma}(r)$ obtain $C^\sigma(rx_k)$, hence $\mathrm{FN}^\sigma(rx_k)$ by IH(a). Have $F(rx_k, l)$, hence $N^\sigma(rx_k, t)$ for some t. Now $N^{\rho\Rightarrow\sigma}(r, \lambda x_k^\rho t)$ by (Ax1).

(b). Assume $\mathrm{FA}^{\rho\Rightarrow\sigma}(r)$ and $C^\rho(s)$. Show $C^\sigma(rs)$. By IH(b) it suffices to show $\mathrm{FA}^\sigma(rs)$. So assume $F(rs,k)$; show $A(rs,t)$ for some t. From $F(rs,k)$ obtain $F(r,k)$ and $F(s,k)$ (axioms on F). Using $\mathrm{FA}(r)$ obtain $A(r,r_1)$ for some r_1. By IH(a) obtain $\mathrm{FN}(s)$, so $N(s,s_1)$ for some s_1. Hence $A(rs, r_1 s_1)$ (axiom on A).

Lemma 1: (a) $C^\rho(r) \to \mathrm{FN}^\rho(r)$, (b) $\mathrm{FA}^\rho(r) \to C^\rho(r)$

Induction on ρ. **Case** ι. (a) holds by definition. (b). Assume $\mathrm{FA}(r)$, that is $\forall k.F(r,k) \to \exists s\, A^\iota(r,s)$. We must show $C^\iota(r)$, that is $\forall k.F(r,k) \to \exists s\, N^\iota(r,s)$. Use (Ax2): $A^\iota(r,s) \to N^\iota(r,s)$.

Case $\rho \Rightarrow \sigma$. (a). Assume $C^{\rho\Rightarrow\sigma}(r)$ and $F(r,k)$. Show $\exists s\, N(r,s)$. Let $\vec{\rho} \vdash r\colon \rho$, $\vec{\sigma} := e(\vec{\rho}, k, \rho)$, $l := \mathrm{Lh}(\vec{\rho}, \vec{\sigma})$. Have $\mathrm{FA}^\rho(x_k)$ by $A^\rho(x_k, x_k)$ (axiom on A), hence $C^\rho(x_k)$ by IH(b). From $C^{\rho\Rightarrow\sigma}(r)$ obtain $C^\sigma(rx_k)$, hence $\mathrm{FN}^\sigma(rx_k)$ by IH(a). Have $F(rx_k, l)$, hence $N^\sigma(rx_k, t)$ for some t. Now $N^{\rho\Rightarrow\sigma}(r, \lambda x_k^\rho t)$ by (Ax1).

(b). Assume $\mathrm{FA}^{\rho\Rightarrow\sigma}(r)$ and $C^\rho(s)$. Show $C^\sigma(rs)$. By IH(b) it suffices to show $\mathrm{FA}^\sigma(rs)$. So assume $F(rs,k)$; show $A(rs,t)$ for some t. From $F(rs,k)$ obtain $F(r,k)$ and $F(s,k)$ (axioms on F). Using $\mathrm{FA}(r)$ obtain $A(r,r_1)$ for some r_1. By IH(a) obtain $\mathrm{FN}(s)$, so $N(s,s_1)$ for some s_1. Hence $A(rs, r_1 s_1)$ (axiom on A).

Lemma 2: $C^\rho(r') \to H(r,r') \to C^\rho(r)$

Proof.
Induction on ρ.

Case ι. Assume $C^\iota(r')$, $H(r,r')$ and $F(r,k)$. Show $N(r,s)$ for some s. Have $F(r',k)$ (axiom on F). Hence $N(r',s)$ for some s. Use $H(r,r') \to N(r',s) \to N(r,s)$ (axiom on H).

Case $\rho \Rightarrow \sigma$. Assume $C^{\rho\Rightarrow\sigma}(r')$, $H(r,r')$ and $C^\rho(s)$. Show $C^\sigma(rs)$. From $C^{\rho\Rightarrow\sigma}(r')$ obtain $C^\sigma(r's)$. Obtain $H(rs,r's)$ from $H(r,r') \to H(rs,r's)$ (axiom on H). Hence $C^\sigma(rs)$ by IH. □

Lemma 2: $C^\rho(r') \to H(r, r') \to C^\rho(r)$

Proof.

Induction on ρ.

Case ι. Assume $C^\iota(r')$, $H(r, r')$ and $F(r, k)$. Show $N(r, s)$ for some s. Have $F(r', k)$ (axiom on F). Hence $N(r', s)$ for some s. Use $H(r, r') \to N(r', s) \to N(r, s)$ (axiom on H).

Case $\rho \Rightarrow \sigma$. Assume $C^{\rho\Rightarrow\sigma}(r')$, $H(r, r')$ and $C^\rho(s)$. Show $C^\sigma(rs)$. From $C^{\rho\Rightarrow\sigma}(r')$ obtain $C^\sigma(r's)$. Obtain $H(rs, r's)$ from $H(r, r') \to H(rs, r's)$ (axiom on H). Hence $C^\sigma(rs)$ by IH. □

Lemma 3: $\vec{C}^{\vec{\rho}}(\vec{s}) \to C^{\rho}(r[\vec{s}])$

$\vec{C}^{\vec{\rho}}(\vec{s})$ means $\mathrm{Lh}(\vec{\rho}) = \mathrm{Lh}(\vec{s}) \wedge \forall i.\, i < \mathrm{Lh}(\vec{\rho}) \to C^{\rho_i}(s_i)$.
"Every term r is computable under substitution".

What is substitution $r[\vec{s}]$? Terms in de Bruijn notation are built from variables k (viewed as indices) by application rs and typed abstraction $\lambda^{\rho} r$. Then we need to consider terms in a context $\rho_0, \ldots, \rho_{k-1}$ (types of all free variables $x_0, \ldots, x_{k-1}$).

Substitution $r[\vec{s}]$ of the first $\mathrm{Lh}(\vec{s})$ variables by $\vec{s}$:

$$k[] := k \qquad (rs)[\vec{s}] := r[\vec{s}]s[\vec{s}]$$
$$0[s :: \vec{s}] := s \qquad (\lambda^{\rho} r)[\vec{s}] := \lambda^{\rho} r[0 :: \vec{s}\uparrow]$$
$$(k+1)[s :: \vec{s}] := k[\vec{s}]$$

Lemma 3: $\vec{C}^{\vec{\rho}}(\vec{s}) \to C^{\rho}(r[\vec{s}])$

$\vec{C}^{\vec{\rho}}(\vec{s})$ means $\mathrm{Lh}(\vec{\rho}) = \mathrm{Lh}(\vec{s}) \wedge \forall i.\, i < \mathrm{Lh}(\vec{\rho}) \to C^{\rho_i}(s_i)$.
"Every term r is computable under substitution".

What is substitution $r[\vec{s}]$? Terms in de Bruijn notation are built from variables k (viewed as indices) by application rs and typed abstraction $\lambda^{\rho} r$. Then we need to consider terms in a context $\rho_0, \ldots, \rho_{k-1}$ (types of all free variables $x_0, \ldots, x_{k-1}$).

Substitution $r[\vec{s}]$ of the first $\mathrm{Lh}(\vec{s})$ variables by $\vec{s}$:

$$k[] := k \qquad (rs)[\vec{s}] := r[\vec{s}]s[\vec{s}]$$
$$0[s :: \vec{s}] := s \qquad (\lambda^{\rho} r)[\vec{s}] := \lambda^{\rho} r[0 :: \vec{s}\uparrow]$$
$$(k+1)[s :: \vec{s}] := k[\vec{s}]$$

Lemma 3: $\vec{C}^{\vec{\rho}}(\vec{s}) \to C^{\rho}(r[\vec{s}])$

$\vec{C}^{\vec{\rho}}(\vec{s})$ means $\mathrm{Lh}(\vec{\rho}) = \mathrm{Lh}(\vec{s}) \wedge \forall i.\, i < \mathrm{Lh}(\vec{\rho}) \to C^{\rho_i}(s_i)$.
"Every term r is computable under substitution".

What is substitution $r[\vec{s}]$? Terms in de Bruijn notation are built from variables k (viewed as indices) by application rs and typed abstraction $\lambda^{\rho} r$. Then we need to consider terms in a context $\rho_0, \ldots, \rho_{k-1}$ (types of all free variables $x_0, \ldots, x_{k-1}$).

Substitution $r[\vec{s}]$ of the first $\mathrm{Lh}(\vec{s})$ variables by $\vec{s}$:

$$k[] := k \qquad (rs)[\vec{s}] := r[\vec{s}]s[\vec{s}]$$
$$0[s :: \vec{s}] := s \qquad (\lambda^{\rho} r)[\vec{s}] := \lambda^{\rho} r[0 :: \vec{s}\uparrow]$$
$$(k+1)[s :: \vec{s}] := k[\vec{s}]$$

Lemma 3: $\vec{C}^{\vec{\rho}}(\vec{s}) \to C^{\rho}(r[\vec{s}])$

$\vec{C}^{\vec{\rho}}(\vec{s})$ means $\mathrm{Lh}(\vec{\rho}) = \mathrm{Lh}(\vec{s}) \wedge \forall i.\, i < \mathrm{Lh}(\vec{\rho}) \to C^{\rho_i}(s_i)$.
"Every term r is computable under substitution".

What is substitution $r[\vec{s}]$? Terms in de Bruijn notation are built from variables k (viewed as indices) by application rs and typed abstraction $\lambda^{\rho} r$. Then we need to consider terms in a context $\rho_0, \ldots, \rho_{k-1}$ (types of all free variables $x_0, \ldots, x_{k-1}$).

Substitution $r[\vec{s}]$ of the first $\mathrm{Lh}(\vec{s})$ variables by $\vec{s}$:

$$k[] := k \qquad (rs)[\vec{s}] := r[\vec{s}]s[\vec{s}]$$
$$0[s :: \vec{s}] := s \qquad (\lambda^{\rho} r)[\vec{s}] := \lambda^{\rho} r[0 :: \vec{s}\uparrow]$$
$$(k+1)[s :: \vec{s}] := k[\vec{s}]$$

Lemma 3: $\vec{C}^{\vec{\rho}}(\vec{s}) \to C^{\rho}(r[\vec{s}])$

$\vec{C}^{\vec{\rho}}(\vec{s})$ means $\mathrm{Lh}(\vec{\rho}) = \mathrm{Lh}(\vec{s}) \wedge \forall i.\, i < \mathrm{Lh}(\vec{\rho}) \to C^{\rho_i}(s_i)$.
"Every term r is computable under substitution".

What is substitution $r[\vec{s}]$? Terms in de Bruijn notation are built from variables k (viewed as indices) by application rs and typed abstraction $\lambda^{\rho} r$. Then we need to consider terms in a context $\rho_0, \ldots, \rho_{k-1}$ (types of all free variables $x_0, \ldots, x_{k-1}$).

Substitution $r[\vec{s}]$ of the first $\mathrm{Lh}(\vec{s})$ variables by $\vec{s}$:

$$k[] := k \qquad (rs)[\vec{s}] := r[\vec{s}]s[\vec{s}]$$
$$0[s :: \vec{s}] := s \qquad (\lambda^{\rho} r)[\vec{s}] := \lambda^{\rho} r[0 :: \vec{s}\uparrow]$$
$$(k+1)[s :: \vec{s}] := k[\vec{s}]$$

Proof of lemma 3: $\vec{C}^{\vec{\rho}}(\vec{s}) \longrightarrow C^{\rho}(r[\vec{s}])$

Induction on r.

Case x_k. Then $C(s_k)$ by assumption.

Case rs. Assume $\vec{C}^{\vec{\rho}}(\vec{s})$. By IH $C^{\rho \Rightarrow \sigma}(r[\vec{s}])$ and $C^{\rho}(s[\vec{s}])$. Hence $C^{\rho}((rs)[\vec{s}])$ by definition of C.

Case $\lambda^{\rho} r$. Assume $\vec{C}^{\vec{\rho}}(\vec{s})$. Show $C^{\rho \Rightarrow \sigma}((\lambda^{\rho} r)[\vec{s}])$. Let $C^{\rho}(s)$. Show $C^{\sigma}((\lambda^{\rho} r)[\vec{s}]s)$. Have $C^{\sigma}(r[s, \vec{s}])$ by IH, and $H((\lambda^{\rho} r)[\vec{s}]s, r[s, \vec{s}])$ (axiom on H). Use lemma 2.

Proof of lemma 3: $\vec{C}^{\vec{\rho}}(\vec{s}) \longrightarrow C^{\rho}(r[\vec{s}])$

Induction on r.

Case x_k. Then $C(s_k)$ by assumption.

Case rs. Assume $\vec{C}^{\vec{\rho}}(\vec{s})$. By IH $C^{\rho \Rightarrow \sigma}(r[\vec{s}])$ and $C^{\rho}(s[\vec{s}])$. Hence $C^{\rho}((rs)[\vec{s}])$ by definition of C.

Case $\lambda^{\rho} r$. Assume $\vec{C}^{\vec{\rho}}(\vec{s})$. Show $C^{\rho \Rightarrow \sigma}((\lambda^{\rho} r)[\vec{s}])$. Let $C^{\rho}(s)$. Show $C^{\sigma}((\lambda^{\rho} r)[\vec{s}]s)$. Have $C^{\sigma}(r[s, \vec{s}])$ by IH, and $H((\lambda^{\rho} r)[\vec{s}]s, r[s, \vec{s}])$ (axiom on H). Use lemma 2.

Normalization Theorem: $\exists s\, N^\rho(r,s)$

Proof.

Have $\mathrm{FA}^{\rho_k}(x_k)$, hence $\vec{C}^{\vec{\rho}}(\vec{x})$ by lemma 1(b). Therefore $C^\rho(r)$ by lemma 3, hence $\mathrm{FN}^\rho(r)$ by lemma 1(a). By $F^\rho(r, \mathrm{Lh}(\vec{\rho}))$ (axiom on F) obtain $\exists s\, N^\rho(r,s)$. □

Normalization Theorem: $\exists s\, N^\rho(r,s)$

Proof.

Have $\mathrm{FA}^{\rho_k}(x_k)$, hence $\vec{C}^{\vec{\rho}}(\vec{x})$ by lemma 1(b). Therefore $C^\rho(r)$ by lemma 3, hence $\mathrm{FN}^\rho(r)$ by lemma 1(a). By $F^\rho(r, \mathrm{Lh}(\vec{\rho}))$ (axiom on F) obtain $\exists s\, N^\rho(r,s)$. □

Predicates

N, A and H are inductively defined by the clauses

$$\mathrm{BetaNf}(r,t) \to \mathrm{EtaExp}^{\rho}_{\vec{\rho}}(t,s) \to N^{\rho}_{\vec{\rho}}(r,s),$$
$$(\vec{\rho} \vdash x_k \colon \rho) \to A^{\rho}_{\vec{\rho}}(x_k,x_k) \quad \text{(writing } x_k \text{ for } k\text{)},$$
$$A^{\rho\Rightarrow\sigma}_{\vec{\rho}}(r,r_1) \to (\vec{\rho} \vdash s \colon \rho) \to N^{\rho}_{\vec{\rho}}(s,s_1) \to A^{\sigma}_{\vec{\rho}}(rs,r_1s_1),$$
$$H\big((\lambda^{\rho} r)[\vec{s}\,]s, r[s,\vec{s}\,]\big).$$

Abbreviations:

$$F^{\rho}_{\vec{\rho}}(r,k) := (\vec{\rho} \vdash r \colon \rho) \land \mathrm{Lh}(\vec{\rho}) \le k,$$
$$\mathrm{FN}^{\rho}_{\vec{\rho}}(r) := \forall k.F^{\rho}_{\vec{\rho}}(r,k) \to \exists s\, N^{\rho}_{\vec{\rho}}(r,s),$$
$$\mathrm{FA}^{\rho}_{\vec{\rho}}(r) := \forall k.F^{\rho}_{\vec{\rho}}(r,k) \to \exists s\, A^{\rho}_{\vec{\rho}}(r,s),$$

Axioms

Usual logical axioms, and induction axioms for the free algebras involved (`boole`, `nat`, `type`, `term`).

Write x_k for k, and $\lambda x_k^\rho\, r$ for $\lambda^\rho r[1, \ldots, k, 0]$.

Ax1. $F_{\vec{\rho}}^{\rho \Rightarrow \sigma}(r, k) \to N_{\vec{\rho}, e(\vec{\rho}, k, \rho)}^{\sigma}(r x_k, s) \to N_{\vec{\rho}}^{\rho \Rightarrow \sigma}(r, \lambda x_k^\rho s)$.

Ax2. $A_{\vec{\rho}}^{\iota}(r, s) \to N_{\vec{\rho}}^{\iota}(r, s)$.

Ax3. $H(r, s) \to N_{\vec{\rho}}^{\rho}(s, t) \to N_{\vec{\rho}}^{\rho}(r, t)$.

Ax4. $H(r, s) \to H(rt, st)$.

Axioms

Usual logical axioms, and induction axioms for the free algebras involved (`boole`, `nat`, `type`, `term`).

Write x_k for k, and $\lambda x^\rho_k\, r$ for $\lambda^\rho r[1, \ldots, k, 0]$.

Ax1. $F^{\rho\Rightarrow\sigma}_{\vec\rho}(r,k) \to N^\sigma_{\vec\rho,e(\vec\rho,k,\rho)}(rx_k, s) \to N^{\rho\Rightarrow\sigma}_{\vec\rho}(r, \lambda x^\rho_k s)$.

Ax2. $A^\iota_{\vec\rho}(r,s) \to N^\iota_{\vec\rho}(r,s)$.

Ax3. $H(r,s) \to N^\rho_{\vec\rho}(s,t) \to N^\rho_{\vec\rho}(r,t)$.

Ax4. $H(r,s) \to H(rt, st)$.

Axioms

Usual logical axioms, and induction axioms for the free algebras involved ($\mathtt{boole}, \mathtt{nat}, \mathtt{type}, \mathtt{term}$).

Write x_k for k, and $\lambda x_k^\rho\, r$ for $\lambda^\rho r[1, \ldots, k, 0]$.

Ax1. $F_{\vec{\rho}}^{\rho \Rightarrow \sigma}(r, k) \to N_{\vec{\rho}, e(\vec{\rho}, k, \rho)}^{\sigma}(r x_k, s) \to N_{\vec{\rho}}^{\rho \Rightarrow \sigma}(r, \lambda x_k^\rho s)$.

Ax2. $A_{\vec{\rho}}^{\iota}(r, s) \to N_{\vec{\rho}}^{\iota}(r, s)$.

Ax3. $H(r, s) \to N_{\vec{\rho}}^{\rho}(s, t) \to N_{\vec{\rho}}^{\rho}(r, t)$.

Ax4. $H(r, s) \to H(rt, st)$.

Computability predicates with explicit realizers

$$\begin{aligned}
\bar{C}^{\iota}_{\vec{\rho}}(a,r) &:= (\vec{\rho} \vdash r : \iota) \wedge P_\iota(a) \wedge \\
&\quad \forall k. F^{\iota}_{\vec{\rho}}(r,k) \to N^{\iota}_{\vec{\rho}}(r, \mathrm{ModIota}(a,k)), \\
\bar{C}^{\rho\Rightarrow\sigma}_{\vec{\rho}}(a,r) &:= (\vec{\rho} \vdash r : \rho \Rightarrow \sigma) \wedge P_{\rho\Rightarrow\sigma}(a) \wedge \\
&\quad \forall \vec{\sigma}, b, s. \bar{C}^{\rho}_{\vec{\rho},\vec{\sigma}}(b,s) \to \bar{C}^{\sigma}_{\vec{\rho},\vec{\sigma}}(\mathrm{Mod}(a,b), rs).
\end{aligned}$$

$\bar{C}^{\rho}_{\vec{\rho}}(a,r)$ has no computational content (because N hasn't).

Then define the computability predicates by

$$C^{\rho}_{\vec{\rho}}(r) := \exists a\, \bar{C}^{\rho}_{\vec{\rho}}(a,r),$$

and prove (the $\forall^{\mathrm{nc}}$-closures of) their properties above:

Computability predicates with explicit realizers

$$\begin{aligned}
\bar{C}^{\iota}_{\vec{\rho}}(a,r) &:= (\vec{\rho} \vdash r : \iota) \wedge P_\iota(a) \wedge \\
&\quad \forall k. F^{\iota}_{\vec{\rho}}(r,k) \to N^{\iota}_{\vec{\rho}}(r, \mathrm{ModIota}(a,k)), \\
\bar{C}^{\rho\Rightarrow\sigma}_{\vec{\rho}}(a,r) &:= (\vec{\rho} \vdash r : \rho \Rightarrow \sigma) \wedge P_{\rho\Rightarrow\sigma}(a) \wedge \\
&\quad \forall \vec{\sigma}, b, s. \bar{C}^{\rho}_{\vec{\rho},\vec{\sigma}}(b,s) \to \bar{C}^{\sigma}_{\vec{\rho},\vec{\sigma}}(\mathrm{Mod}(a,b), rs).
\end{aligned}$$

$\bar{C}^{\rho}_{\vec{\rho}}(a,r)$ has no computational content (because N hasn't).

Then define the computability predicates by

$$C^{\rho}_{\vec{\rho}}(r) := \exists a\, \bar{C}^{\rho}_{\vec{\rho}}(a,r),$$

and prove (the $\forall^{\mathrm{nc}}$-closures of) their properties above:

Computability predicates with explicit realizers

$$\begin{aligned}
\bar{C}^{\iota}_{\vec{\rho}}(a,r) &:= (\vec{\rho} \vdash r : \iota) \wedge P_{\iota}(a) \wedge \\
&\quad \forall k. F^{\iota}_{\vec{\rho}}(r,k) \to N^{\iota}_{\vec{\rho}}(r, \mathrm{ModIota}(a,k)), \\
\bar{C}^{\rho\Rightarrow\sigma}_{\vec{\rho}}(a,r) &:= (\vec{\rho} \vdash r : \rho \Rightarrow \sigma) \wedge P_{\rho\Rightarrow\sigma}(a) \wedge \\
&\quad \forall \vec{\sigma}, b, s. \bar{C}^{\rho}_{\vec{\rho},\vec{\sigma}}(b,s) \to \bar{C}^{\sigma}_{\vec{\rho},\vec{\sigma}}(\mathrm{Mod}(a,b), rs).
\end{aligned}$$

$\bar{C}^{\rho}_{\vec{\rho}}(a,r)$ has no computational content (because N hasn't).

Then define the computability predicates by

$$C^{\rho}_{\vec{\rho}}(r) := \exists a\, \bar{C}^{\rho}_{\vec{\rho}}(a,r),$$

and prove (the $\forall^{\mathrm{nc}}$-closures of) their properties above:

Computability predicates

Lemma

$$C^{\iota}_{\vec{\rho}}(r) \quad \leftrightarrow (\vec{\rho} \vdash r : \iota) \wedge \mathrm{FN}^{\iota}_{\vec{\rho}}(r),$$
$$C^{\rho \Rightarrow \sigma}_{\vec{\rho}}(r) \leftrightarrow (\vec{\rho} \vdash r : \rho \Rightarrow \sigma) \wedge \forall^{\mathrm{nc}} s, \vec{\sigma}. C^{\rho}_{\vec{\rho},\vec{\sigma}}(s) \to C^{\sigma}_{\vec{\rho},\vec{\sigma}}(rs).$$

The proof uses

(AC) $\forall x \exists y\, A(x,y) \to \exists f \forall x\, A(x, f(x))$ $(A(x,y)$ arbitrary),

(IP) $(A \to \exists x\, B(x)) \to \exists x.A \to B(x)$, where $\tau(A) = \varepsilon$,

(UNC) $\forall^{\mathrm{nc}} x \exists y\, A(x,y) \to \exists y \forall^{\mathrm{nc}} x\, A(x,y)$,

which are realized by identity functions.

Extracted term: lemma 1

```
(Rec type=>(omega=>nat=>term)@@((nat=>term)=>omega))
(ModIota@OmegaInIota)
([rho3,rho4,p5,p6]
  ([a7,n8]
    Abs rho3
    (Sub
     (left p6(Mod a7(right p5([n9]Var n8)))(Succ n8))
     (Wrap(Succ(Succ n8))
      ((Var map Seq 1 n8):+:(Var 0):))))@
  ([g7]
    Hat rho3 rho4
    ([a8]right p6([n9]g7 n9(left p5 a8 n9)))))
```

Lemma 1 $\sim$ reify & reflect

Disregarding administrative functions and translating via

`rho4`	`rho5`	`left p5`	`right p5`	`left p6`	`right p6`
ρ	σ	$\downarrow_\rho$	$\uparrow_\rho$	$\downarrow_\sigma$	$\uparrow_\sigma$

gives

$$\downarrow_\rho : \mathbf{C}_\omega \to (\mathbf{N} \to \mathbf{\Lambda}) \ (\text{“reify”}) \quad \uparrow_\rho : (\mathbf{N} \to \mathbf{\Lambda}) \to \mathbf{C}_\omega \ (\text{“reflect”}),$$

with the recursion equations

$$\downarrow_\iota(r) := r, \qquad \uparrow_\iota(r) := r,$$
$$\downarrow_{\rho\Rightarrow\sigma}(a)(k) := \lambda x^\rho_k.\downarrow_\sigma\big(a(\uparrow_\rho(x^\infty_k))\big)(k+1), \quad \uparrow_{\rho\Rightarrow\sigma}(r)(b) := \uparrow_\sigma(r\downarrow_\rho(b)).$$

Extracted term: lemma 3

```
(Rec term=>list type=>list omega=>omega)
([n3,rhos4](ListProj omega)n3)
([r3,r4,q5,q6,rhos7,as8]
  Mod(q5 rhos7 as8)(q6 rhos7 as8))
([rho3,r4,q5,rhos6,as7]
  Hat rho3(Typ(rho3::rhos6)r4)
  ([a8]cLemmaTwo(q5(rho3::rhos6)(a8::as7))))
```

Lemma 3 $\sim$ evaluation

For $\mathtt{cLemmaThree}(r, \vec{\rho}, \vec{a})$ write

$$[\![r]\!]_{(x_0^{\rho_0} \mapsto a_0, \ldots, x_{k-1}^{\rho_{k-1}} \mapsto a_{k-1})} \quad \text{with } k := \mathrm{Lh}(\vec{\rho})$$

or $[\![r]\!]_{(\vec{x} \mapsto \vec{a})}$. Disregarding administrative functions gives

$$\begin{aligned}
[\![x_i]\!]_{(\vec{x} \mapsto \vec{a})} &= a_i \\
[\![rs]\!]_{(\vec{x} \mapsto \vec{a})} &= [\![r]\!]_{(\vec{x} \mapsto \vec{a})}[\![s]\!]_{(\vec{x} \mapsto \vec{a})} \\
[\![\lambda x_k^{\rho}\, r]\!]_{(\vec{x} \mapsto \vec{a})}(b) &= [\![r]\!]_{(\vec{x}, x_k^{\rho} \mapsto \vec{a}, b)}
\end{aligned}$$

NThm $\sim$ normalization by evaluation

Extracted term: NThm:

```
[rhos0,r1]
 left(cLemmaOne(Typ rhos0 r1))
 (cLemmaThree r1 rhos0(cSCrsSeq rhos0(Nil type)))
 Lh rhos0
```

Let $\uparrow$ denote the variable assignment $x_k^\rho \mapsto \uparrow_\rho(x_k)$. Then $\mathtt{cNThm}(\vec{\rho}, r)$ computes the long normal form of r as

$$\downarrow_\rho([\![r]\!]_\uparrow)(k) \quad \text{with } k = \mathrm{Lh}(\vec{\rho}).$$

This is "normalization by evaluation".

Conclusion

- Program extraction from proofs not only gives certified code ("no logical errors"), but (in case of clever proofs) can even give unexpected algorithms.
- A distinction between universal quantifiers with and without computational content is necessary. (The same applies to inductively defined predicates.)
- Partial continuous functionals as intended domains can be indispensible for an appropriate formulation.
- Future work: "Type theory with approximations".