On Normalization by Evaluation for Object Calculi

Jan Schwinghammer

Programming Systems Lab, Saarland University, Saarbrücken, Germany

Abstract. We present a procedure for computing normal forms of terms in Abadi and Cardelli's functional object calculus. Even when equipped with simple types, terms of this calculus are not terminating in general, and we draw on recent ideas about the *normalization by evaluation* paradigm for the untyped lambda calculus. Technically, we work in the framework of Shinwell and Pitts' FM-domain theory, which leads to a normalization procedure for the object calculus that is directly implementable in a language like Fresh O'Caml.

1 Introduction

Normalization by evaluation (NBE), sometimes referred to as *reduction-free normalization*, is a technique for computing normal forms of terms. It was proposed in [7] as an efficient method for proof normalization, based on the representation of natural deduction proofs as terms of the simply typed lambda calculus (possibly enriched with constants). The underlying principle, once discovered, is rather simple: In a model of the calculus, the denotations $[\![a]\!]$ and $[\![a']\!]$ of any convertible terms $a \leftrightarrow a'$ are necessarily identified, so if it is possible to *extract* a normal form b representing a semantic element d (i.e., such that $[\![b]\!] = d$) then interpretation followed by extraction yields a normal form for any given term.

Of course, the trick is to find models that allow for such term extraction. *Residualizing models* contain (representations of) syntax and provide basic operations on terms, which can be used to construct normal forms inside the model. For instance, a residualizing interpretation of simply typed lambda calculus may be obtained by constructing the full set-theoretic hierarchy over the set of terms, where a term extraction function $\downarrow$, along with a term embedding function $\uparrow$ from lambda terms to semantic elements, can be defined by mutual induction on the type: letting $\downarrow^{\iota}(a) = a$ and $\uparrow^{\iota}(a) = a$ at base types ι, one considers

$$\downarrow^{A\Rightarrow B}(f) = \mathsf{lam}(x, \downarrow^{B}(f(\uparrow^{A}(\mathsf{var}\,x)))) \tag{1}$$

$$\uparrow^{A\Rightarrow B}(a) = \lambda(v \in [\![A]\!]).\ \uparrow^{B}(\mathsf{app}(a, \downarrow^{A}(v)) \tag{2}$$

where lam, app and var are used as constructors for lambda terms. Indeed, [7] proves that, as long as the variable x in (1) is chosen 'fresh', if ρ is the identity environment then $\downarrow^{A}([\![a]\!]_{\rho})$ is a $\beta\eta$-long normal form of the term $a : A$.

If one uses a (functional) programming language as an adequate meta-language to describe the interpretation and extraction of terms, then NBE leads to a normalization method that is directly executable, and correct by construction. The method is robust and widely applicable; it has been adapted to various type theories, going well beyond simply typed lambda calculus (for instance, see [4, 3, 8, 7]). An introductory survey, including applications of NBE to partial evaluation of functional programs, can be found in [10].

In this paper, we present a NBE procedure for Abadi and Cardelli's functional object calculus [1]. Even when equipped with simple types, the terms of this calculus are not in general convertible to normal forms. Recursion is inherent to objects, and arises from the presence of a distinguished 'self' identifier that provides access to the host object from within method bodies. (Terms are also not necessarily normalizing in similar foundational calculi for object-oriented languages, e.g. [12], and we expect that our results can be straightforwardly adapted.) Thus, we draw heavily on work about NBE of *untyped* lambda calculus [11, 2]. In hindsight this is not surprising, since indeed it bears many similarities to the self-application model of method invocation in object calculi. To take into account the partiality of the normalization function, the correctness criterion is weakened accordingly [11]: it comprises of *soundness* (results are normal forms of the respective input), *identification* (convertible inputs yield the same result), and *completeness* (the function is defined on every input that is convertible to a normal form), which we also establish for our NBE procedure.

Just as in the defining equation (1) for abstractions above, a technical complication arises during the extraction of terms corresponding to the methods of an object in normal form. One has to find variables that are fresh, not simply with respect to some given term, but rather 'globally': in general the name of the bound variable must be chosen *before* the term for the method body can be constructed recursively. Most previous work has addressed this issue by guaranteeing that variables generated in the normalization process are indeed globally unique, for instance, by implementing a name generator using a state monad, or avoided name clashes by adopting de Bruijn levels to identify variables. In contrast, we follow Pitts and construct a residualizing model using *nominal sets*, which allow for a rigorous yet fairly lightweight treatment of binding constructs, via built-in notions of *finite support* and *freshness* [16].

More precisely, our model for the object calculus is a variant of the untyped domain model of [17], where (1) we replace the category of domains by the FM-domain theory of Shinwell and Pitts, and (2) use a continuation semantics in the style of Shinwell and Pitts, and Benton and Leperchey [18, 19, 6]. This provides a neat solution to the problem of interpreting fresh name generation in the meta-language, and allows for a conceptually clear presentation of NBE and its correctness. While the overall structure of our proof closely follows that of Filinski and Rohde's [11], proofs of individual properties have a distinct flavour due to the continuation semantics. In particular, the central relation between denotations and syntax is an instance of the relational $\top\top$-lifting of [14, 19].

Outline The next section recalls Abadi and Cardelli's calculus. Section 3 summarizes the relevant aspects from FM domain theory. The construction of the

$$\frac{}{x : \mathsf{at}} \qquad \frac{a : \mathsf{at}}{a.\ell : \mathsf{at}} \qquad \frac{a : \mathsf{at} \qquad m : \mathsf{mnf}}{a.\ell := m : \mathsf{at}} \qquad \frac{(\forall i \in [k])\ m_i : \mathsf{mnf}}{[\ell_i = m_i{}^{i \in [k]}] : \mathsf{nf}} \qquad \frac{a : \mathsf{at}}{a : \mathsf{nf}} \qquad \frac{a : \mathsf{nf}}{\varsigma(x)a : \mathsf{mnf}}$$

Fig. 1. Atomic terms and normal forms

If $a \equiv [\ell_i = m_i{}^{i \in [k]}]$, $j \in [k]$ and $m_j \equiv \varsigma(x_j)a_j$ then
Selection: $a.\ell_j \ \leftrightarrow \ (x_j \mapsto a)(a_j)$
Update: $a.\ell_j := m \ \leftrightarrow \ [\ell_i = m_i{}^{i \in [k], i \neq j}, \ell_j = m]$

Fig. 2. Conversion semantics

normalization procedure is given in Section 4, its correctness is established in Section 5 which forms the technical core of the paper. The appendix contains the existence proof omitted in Section 5.

2 Syntax and Conversion Semantics of Object Calculus

Syntax Fix a set $\mathcal{L}$ of *labels*, ranged over by ℓ, and let x, y range over a countably infinite set of variables *Var*. For $k \in \mathbb{N}$, let $[k] = \{1, \ldots, k\}$. The set *obj* of object calculus terms is defined by the following grammar:

$$a, b \in \mathit{obj} ::= x \mid [\ell_i = m_i{}^{i \in [k]}] \mid a.\ell \mid a.\ell := m$$
$$m \in \mathit{meth} ::= \varsigma(x)a$$

The self binder of methods is the only binding construct ($\varsigma(x)a$ binds x in the method body a). This determines the set $fv(a)$ of free variables of a, and we identify terms up to α-equivalence. Given a (finite) map $\theta : \mathit{Var} \to \mathit{obj}$ we write capture-avoiding simultaneous substitution as $\theta(a)$.

The set of *atomic* (or *neutral*) *terms* and *(method) normal forms*, respectively, are defined inductively by the inference rules in Figure 1. These will be the output of the nbe procedure (if any). They roughly correspond to the 'well-formed' irreducible terms with respect to the usual reduction semantics from [1] (informally called *results* in [1]). In particular, note that $[\ell_i = \varsigma(x_i)a_i{}^{i \in [k]}].\ell$ and $[\ell_i = \varsigma(x_i)a_i{}^{i \in [k]}].\ell := \varsigma(y)b$ are not normal forms whenever $\ell \neq \ell_i$ for all $i \in [k]$ even though both are irreducible. It is not difficult to repair this mismatch by considering a minor variation of the reduction relation; e.g. as in [9].

Conversion The conversion relation $a \leftrightarrow b$ between terms is the least equivalence relation on *obj* containing the axioms in Figure 2 that is compatible, i.e., it is reflexive, symmetric, transitive, and for all contexts $C[-]$ with a single hole, if $a \leftrightarrow b$ then $C[a] \leftrightarrow C[b]$.

3 FM Domain Theory

We work in the category **FM-Cppo** of FM-cppos over *Var* and equivariant strict continuous functions. To keep this paper self-contained we recall the necessary definitions from [19, 18]. In the following we write $(x\,x')$ for the transposition that swaps x and x' and fixes all other $y \in \mathit{Var}$, and denote by $\mathit{perm} = \mathit{perm}(\mathit{Var})$ the group of bijections $\pi : \mathit{Var} \to \mathit{Var}$ generated by all transpositions (i.e., where $\pi(x) = x$ for all but finitely many $x \in \mathit{Var}$); id stands for the identity. An action of perm on a set A is an operation $\cdot : \mathit{perm} \times A \to A$ such that $\mathit{id} \cdot a = a$ and $\pi \cdot (\pi' \cdot a) = (\pi \circ \pi') \cdot a$ for all $a \in A$ and $\pi, \pi' \in \mathit{perm}$. An *FM-set* A is given by an action of perm on A such that every $a \in A$ is *finitely supported*, meaning that there exist finite sets $V \subseteq \mathit{Var}$ such that for all $\pi \in \mathit{perm}$,

$$\big(\forall (x \in V)\ \pi(x) = x\big) \quad \Rightarrow \quad \pi \cdot a = a.$$

Each $a \in A$ in fact possesses a smallest such set $\mathit{supp}(a)$ supporting a. Every set A gives rise to an FM-set when equipped with the trivial action $\pi \cdot a = a$.

Syntax as FM-Set The set of *obj*-terms may be turned into an FM-set, where the action of perm is defined by structural recursion and the interesting cases are $\pi \cdot x = \pi(x)$ and $\pi \cdot \varsigma(x)a = \varsigma(\pi(x))(\pi \cdot a)$. This gives rise to a well-defined action of perm on α-equivalence classes of terms such that the notion of support coincides with that of free variables, i.e., $\mathit{supp}(a) = \mathit{fv}(a)$ for all $a \in \mathit{obj}$. Moreover, α-equivalence itself can be characterised using the action on terms, as the least congruence relation $\sim$ such that whenever $(x\,y) \cdot a \sim (x'\,y) \cdot a'$ for some/all y not occurring in a, a' and different from x and x', then $\varsigma(x)a \sim \varsigma(x')a'$ [16].

Domain-theoretic Constructions An *FM-poset* is an FM-set A equipped with a partial order $\sqsubseteq$ on A that is compatible with the group action, i.e. where $a \sqsubseteq a'$ implies $\pi \cdot a \sqsubseteq \pi \cdot a'$ for all a, a' and π. More generally, a subset $D \subseteq A$ of an FM-set A is finitely supported if there exists finite $V \subseteq \mathit{Var}$ such that for all π that fix V pointwise, $a \in D \;\Leftrightarrow\; \pi \cdot a \in D$. By an *FM-cpo* we mean an FM-poset A where each finitely supported directed subset D has a least upper bound $\sqcup D$. (The use of directed-complete FM-cpos is not essential for our purposes; we could just as well have used chain-complete FM-posets as in [19].) A continuous function $f : A \to B$ between FM-cpos is a monotonic function f from A to B such that (1) f is *equivariant*: $f(\pi \cdot a) = \pi \cdot (f(a))$ for all a and π, and (2) f preserves least upper bounds: $f(\sqcup D) = \sqcup f(D)$ for all finitely supported directed sets $D \subseteq A$. An *FM-cppo* is an FM-cpo that possesses a least element $\perp$, for which necessarily $\mathit{supp}(\perp) = \emptyset$ holds. A continuous function f between FM-cppos is *strict* if $f(\perp) = \perp$.

The *smash product* $A_1 \otimes A_2$ and *coalesced sum* $A_1 \oplus A_2$ of FM-cppos A_1 and A_2 are given by the corresponding construction on pointed cpos (e.g. [15]) where perm acts by $\pi \cdot \langle a, a' \rangle = \langle \pi \cdot a, \pi \cdot a' \rangle$ and $\pi \cdot \iota_i(a) = \iota_i(\pi \cdot a)$ for $i = 1, 2$, resp. (We may omit the tags ι_i below). If A is a FM-cpo then its *lift* $A_\perp$ is obtained by adjoining a new element $\perp \notin A$, with $\pi \cdot \perp = \perp$ and $\perp \sqsubseteq a$ for all π, a; conversely, for an FM-cppo, $A_\downarrow$ is the FM-cpo obtained by removing $\perp$.

The *function space* $A \to B$ between FM-cpos consists of those monotonic functions $f : A \to B$ that preserve least upper bounds of finitely supported directed subsets of A, and additionally are finitely supported in the sense that there exists a finite set $V \subseteq \mathit{Var}$ such that for all $\pi \in \mathit{perm}$ and $a \in A$,

$$\big(\forall (x \in V)\ \pi(x) = x\big) \quad \Rightarrow \quad \pi \cdot (f(a)) = f(\pi \cdot a).$$

If A, B are FM-cppos then the *strict function space* $A \multimap B$ is the subset of functions in $A \to B$ that additionally preserve $\perp$. When equipped with the action where $(\pi \cdot f)(a) = \pi \cdot (f(\pi^{-1} \cdot a))$ for $a \in A$, both sets become FM-c(p)pos.

For an FM-cppo A we also consider an FM-cppo of $\mathcal{L}$-labelled records with entries from A: its underlying set is

$$\mathsf{Rec}_{\mathcal{L}}(A) \;=\; \Big(\sum_{L \subseteq_{\mathit{fin}} \mathcal{L}} (L \to A_{\downarrow})\Big)_{\perp} \tag{3}$$

For $\perp \neq \iota_L(r) \in \mathsf{Rec}_{\mathcal{L}}(A)$ we write $\mathsf{dom}(r) = L$ and use record notation $\{\!|\ell_1 = a_1, \ldots, \ell_k = a_k|\!\}$ if $L = \{\ell_1, \ldots, \ell_k\}$ and $r(\ell_i) = a_i$ for all $i \in [k]$. We shall also write $r.\ell$ for $r(\ell)$ and $r.\ell := a$ for the record that maps ℓ to a and all other $\ell' \in \mathsf{dom}(r)$ to $r.\ell'$ (assuming $\ell \in \mathsf{dom}(r)$; the expressions denote $\perp$ otherwise). The ordering on (3) is given by

$$r \sqsubseteq r' \quad \Leftrightarrow \quad r \neq \perp \;\Rightarrow\; \mathsf{dom}(r) = \mathsf{dom}(r') \;\wedge\; \forall (\ell \in \mathsf{dom}(r))\ r.\ell \sqsubseteq r'.\ell,$$

and the action of *perm* by $(\pi \cdot r).\ell = \pi \cdot (r.\ell)$ for all $\ell \in \mathsf{dom}(r)$. In particular, $\mathit{supp}(r) = \bigcup_{\ell \in \mathsf{dom}(r)} \mathit{supp}(r.\ell)$ so that r is finitely supported.

Continuation Monad For the purposes of giving a denotation to objects in a continuation semantics we denote by $A^{\perp}$ the FM-cppo $A \multimap 1_{\perp}$ and by $A^{\perp\perp}$ the FM-cppo $(A^{\perp})^{\perp} = (A \multimap 1_{\perp}) \multimap 1_{\perp}$, where 1 is a singleton cppo.

We shall write $\textsf{\textbf{return}}\ a$ for the unit $\lambda(h \in A^{\perp}).\, h(a)$ of the continuation monad, and denote by $f^* \in (A^{\perp\perp} \multimap B^{\perp\perp})$ the extension

$$f^*(d)(h) = d(\lambda(a \in A).\, f(a)(h))$$

of a function $f \in (A \multimap B^{\perp\perp})$. The notation $\textsf{\textbf{let}}\ a \Leftarrow d\ \textsf{\textbf{in}}\ e[a]$, where e may depend strict continuously on a, stands for $(\lambda(a \in A).\, e[a])^*(d)$. Note that $\textsf{\textbf{return}}$ and $(-)^*$ are equivariant and (strict) continuous operations, and that we have:

$$\begin{aligned} \textsf{\textbf{let}}\ a \Leftarrow \textsf{\textbf{return}}\ a'\ \textsf{\textbf{in}}\ e[a] \;&=\; e[a'] \\ \textsf{\textbf{let}}\ a \Leftarrow (\textsf{\textbf{let}}\ a' \Leftarrow d\ \textsf{\textbf{in}}\ e'[a'])\ \textsf{\textbf{in}}\ e[a] \;&=\; \textsf{\textbf{let}}\ a' \Leftarrow d\ \textsf{\textbf{in}}\ \textsf{\textbf{let}}\ a \Leftarrow e'[a']\ \textsf{\textbf{in}}\ e[a]. \end{aligned}$$

A Domain Equation for Objects An account of the self-application inherent in Abadi and Cardelli's object calculus requires a recursively defined domain. As outlined in [19], the constructions on FM-cppos are functorial, with a *locally FM-continuous* action on the morphisms of **FM-Cppo**. Essentially, this means that solutions to recursive domain equations can be found by the classical technique using embedding-projection pairs, suitably adapted to FM-cppos by replacing arbitrary directed sets in the construction by finitely supported ones.

Similar to the untyped lambda calculus, for the object calculus we will be interested in a model where the space of 'records of pre-methods' is a retract of the model. More precisely, given an FM-cppo A, we let $F_A : \mathbf{FM\text{-}Cppo}^{op} \times \mathbf{FM\text{-}Cppo} \longrightarrow \mathbf{FM\text{-}Cppo}$ be the locally FM-continuous functor

$$F_A(X, Y) = A \oplus \mathsf{Rec}_{\mathcal{L}}(X \multimap Y^{\perp\perp}) \tag{4}$$

and observe that the construction referred to above yields solutions D with $i : F_A(D, D) \cong D$ that are *minimal invariant* objects in the sense that the map $\delta : (D \multimap D) \to (D \multimap D)$, defined as $\delta(e) = i \circ F_A(e, e) \circ i^{-1}$, satisfies $id_D = lfp(\delta)$. This minimal invariant property will be employed in the existence proof in Section 5 below.

For ease of notation we will usually omit the isomorphism i in the following.

4 Normalization Procedure

We will interpret the object calculus in a residualizing model, specified by the following pair of mutually recursive domain equations:

$$\mathbb{O} = obj_\perp \oplus \mathsf{Rec}_{\mathcal{L}}(\mathbb{M}) \qquad\qquad \mathbb{M} = \mathbb{O} \multimap \mathbb{O}^{\perp\perp}$$

Clearly $\mathbb{O}$ can be obtained as the minimal invariant of (4) by choosing $A = obj_\perp$.

Term Constructors The embedding $Var \hookrightarrow obj$ extends to a strict continuous function $\mathsf{var} \in (Var_\perp \multimap obj_\perp)$ with empty support, mapping $x \in Var$ to x. Similarly, the other ways of constructing obj terms may be viewed as strict continuous functions (with empty support):

- $\mathsf{meth} : Var_\perp \otimes obj_\perp \multimap meth_\perp$ sends $\langle x, a\rangle \neq \perp$ to $\varsigma(x)a$
- $\mathsf{obj} : \mathsf{Rec}_{\mathcal{L}}(meth_\perp) \multimap obj_\perp$ sends $\{\!|\ell_i = m_i{}^{i\in[k]}|\!\}$ to $[\ell_i = m_i{}^{i\in[k]}]$
- $\mathsf{sel} : obj_\perp \otimes \mathcal{L}_\perp \multimap obj_\perp$ sends $\langle a, \ell\rangle \neq \perp$ to $a.\ell$
- $\mathsf{upd} : obj_\perp \otimes \mathcal{L}_\perp \otimes meth_\perp \multimap obj_\perp$ sends $\langle a, \ell, m\rangle \neq \perp$ to $a.\ell := m$

If we let $\mathsf{tm}(a) = i(\iota_1(a))$ and $\mathsf{rec}(r) = i(\iota_2(r))$ then any element $\perp \neq d \in \mathbb{O}$ may be uniquely written as either $d = \mathsf{tm}(a)$ or $d = \mathsf{rec}\ r$ for (uniquely determined) $a \in obj$ and $\perp \neq r \in \mathsf{Rec}_{\mathcal{L}}(\mathbb{M})$, respectively.

Reification and Reflection The reason for using FM domain theory and the continuation semantics is that it lets us choose fresh variable names: there exists an element *fresh* in the FM-cppo $(Var_\perp)^{\perp\perp}$ that maps $h \in (Var_\perp)^\perp$ to $h(x) \in 1_\perp$, where x is any variable not in $supp(h)$. The choice of x does not matter since the action of $perm$ on $1_\perp$ is necessarily trivial, hence if $x, y \notin supp(h)$ then $h(x) = ((x\,y) \cdot h)(x) = (x\,y) \cdot (h(y)) = h(y)$.

Now *reflection* $\uparrow : obj_\perp \multimap \mathbb{O}^{\perp\perp}$, $\uparrow(a) = \mathbf{return}(\mathsf{tm}\ a)$, lets us view terms as elements of $\mathbb{O}$. (Conceptually, it would be enough to have reflection for atomic terms only.) Conversely, the (mutually recursive) *reification* functions $\downarrow : \mathbb{O} \multimap$

$$\begin{aligned}
[\![x]\!]_\eta &= \textbf{return}(\eta(x)) \\
\Big[\!\!\Big[[\ell_i = m_i{}^{i\in[k]}]\Big]\!\!\Big]_\eta &= \textbf{let } (f_i \Leftarrow [\![m_i]\!]_\eta \mid i \in [k]) \textbf{ in return}(\mathsf{rec}(\,\{\!|\ell_i = f_i{}^{i\in[k]}|\!\}\,)) \\
[\![a.\ell]\!]_\eta &= \textbf{match } [\![a]\!]_\eta \textbf{ with} \\
&\qquad tm(b) \Rightarrow \uparrow(\mathsf{sel}(b,\ell)) \\
&\qquad \mid rec(r) \Rightarrow r.\ell(\mathsf{rec}\ r) \\
[\![a.l := m]\!]_\eta &= \textbf{match } [\![a]\!]_\eta \textbf{ with} \\
&\qquad tm(b) \Rightarrow \textbf{let } m' \Leftarrow \downarrow_m{}^*([\![m]\!]_\eta) \textbf{ in } \uparrow(\mathsf{upd}(b,\ell,m')) \\
&\qquad \mid rec(r) \Rightarrow \textbf{let } f \Leftarrow [\![m]\!]_\eta \textbf{ in return}(\mathsf{rec}(r.\ell := f)) \\
[\![\varsigma(x)a]\!]_\eta &= \textbf{return}(\lambda(d \in \mathbb{O}).\, \textbf{if } d{=}\bot \textbf{ then } \bot \textbf{ else } [\![a]\!]_{\eta[x:=d]})
\end{aligned}$$

Fig. 3. Interpretation of *obj*-terms in $\mathbb{O}^{\perp\perp}$

$(obj_\perp)^{\perp\perp}$ and $\downarrow_m : \mathbb{M} \multimap (meth_\perp)^{\perp\perp}$, allow us to read back object calculus terms from semantic elements. They are defined as least fixed points, by the equations

$$\begin{aligned}
\downarrow(\bot) &= \bot \\
\downarrow(\mathsf{tm}\ a) &= \textbf{return}(a) \\
\downarrow(\mathsf{rec}\ r) &= \textbf{let } (m_\ell \Leftarrow \downarrow_m(r.\ell) \mid \ell \in \mathsf{dom}(r)) \textbf{ in return}\big(\mathsf{obj}(\,\{\!|\ell = m_\ell{}^{\ell\in\mathsf{dom}(r)}|\!\}\,)\big) \\
\downarrow_m(f) &= \textbf{let } x \Leftarrow \mathit{fresh} \textbf{ in let } a \Leftarrow \downarrow^*(f^*(\uparrow(\mathsf{var}\ x))) \textbf{ in return}\big(\mathsf{meth}(x,a)\big)
\end{aligned}$$

where $\downarrow_m$ makes use of the function *fresh* described above.

Interpretation and Normalization of Objects We interpret *obj* in $\mathbb{O}$. More precisely, given an environment $\eta \in \mathit{Env} = \mathit{Var} {\rightarrow} \mathbb{O}$ such that $\eta(x) \neq \bot$ for all x, the denotation of each term $a \in obj$ is an element $[\![a]\!]_\eta \in \mathbb{O}^{\perp\perp}$. Similarly, the denotation of each method $m \in meth$ is an element $[\![m]\!]_\eta \in \mathbb{M}^{\perp\perp}$. The notation **match** d **with** $tm(a) \Rightarrow e_1[a] \mid rec(r) \Rightarrow e_2[r]$ (where e_1 and e_2 may depend strict continuously on a and r, resp.) stands for the case construct

$$\textbf{let } v \Leftarrow d \textbf{ in } \big((\lambda(a \in obj_\perp).\, e_1) \oplus (\lambda(r \in \mathsf{Rec}_{\mathcal{L}}(\mathbb{M})).\, e_2)\big)(v),$$

so that in particular

$$\textbf{match } (\textbf{return } d) \textbf{ with } tm(a) \Rightarrow e_1[a] \mid rec(r) \Rightarrow e_2[r] = \begin{cases} \bot & \text{if } d = \bot \\ e_1[a] & \text{if } d = \mathsf{tm}\ a \\ e_2[r] & \text{if } d = \mathsf{rec}\ r \end{cases}$$

The interpretation is given in Figure 3, defined by recursion on *obj*-terms. It can be verified that $[\![-]\!]_\eta$ respects α-equivalence and therefore is also well-defined on α-equivalence classes. Alternatively, $[\![-]\!]_\eta$ may be directly defined on equivalence classes by *α-structural recursion* [16].

Lemma 1 (Substitution). *For all a, $\boldsymbol{x} = x_1 \dots x_n$, $\boldsymbol{b} = b_1 \dots b_n$, $\boldsymbol{d} = d_1 \dots d_n$ and η, if $[\![b_i]\!]_\eta = \textbf{return}(d_i)$ for all $i \in [n]$ then $[\![(\boldsymbol{x} \mapsto \boldsymbol{b})(a)]\!]_\eta = [\![a]\!]_{\eta[\boldsymbol{x}:=\boldsymbol{d}]}$.*

Proof. By α-structural induction on a, exploiting that the denotation of a only depends on the value of the environment on $fv(a) = supp(a)$ which is similarly proven by α-structural induction using the definition of $a \mapsto [\![a]\!]_\eta$. See [16]. $\square$

Theorem 1 (Model soundness). *If $a \leftrightarrow a'$ then $[\![a]\!] = [\![a']\!]$.*

Proof (sketch). By induction on the derivation of $a \leftrightarrow a'$. For instance, we have for all a of the form $[\ell_i = m_i{}^{i \in [k]}]$ and $j \in [k]$ with each m_i of the form $\varsigma(x_i)a_i$:

$$\begin{aligned}
[\![a.\ell_j]\!]_\eta &= \textbf{match } [\![a]\!]_\eta \textbf{ with } tm(b) \Rightarrow \uparrow(\mathsf{sel}(b, \ell)) \\
&\qquad\qquad\qquad\quad \mid rec(r) \Rightarrow r.\ell(\mathsf{rec}\ r) \\
&= g_j(\mathsf{rec}\ \{\!|\ell_i = g_i{}^{i \in [k]}|\!\}) \qquad (\text{for } g_i = \lambda d.\textbf{if } d{=}\bot \textbf{ then} \perp \textbf{else } [\![a_i]\!]_{\eta[x_i:=d]}) \\
&= [\![a_j]\!]_{\eta[x_j:=\mathsf{rec}\ \{\!|\ell_i=g_i{}^{i\in[k]}|\!\}\,]} \\
&= [\![(x_j \mapsto a)a_j]\!]_\eta
\end{aligned}$$

where the second equation follows from $[\![a]\!]_\eta = \textbf{return}(\mathsf{rec}\ \{\!|\ell_i = g_i{}^{i \in [k]}|\!\})$ for $g_i = \lambda d.\textbf{if } d{=}\bot \textbf{ then} \perp \textbf{else } [\![a_i]\!]_{\eta[x_i:=d]}$, the second is by definition of g_j, and the last equation is by Lemma 1. The case for update is similar, the cases for the equivalence and compatibility rules are immediate by induction. $\square$

Note that **return** is injective. Thus we may define $norm : obj \to obj_\bot$ to be the partial map satisfying

$$norm(a) = b \quad \Leftrightarrow \quad \downarrow^*([\![a]\!]_{\eta_0}) = \textbf{return}(b) \qquad (\text{some } b \in obj) \tag{5}$$

and $norm(a) = \bot$ otherwise. Here, $\eta_0 = \lambda(x \in Var).\,\mathsf{tm}(\mathsf{var}\ x)$ denotes the environment that maps every variable to the corresponding element of $\mathbb{O}$.

5 Correctness

Following [11], the correctness properties we expect from $norm : obj \to obj_\bot$ are split into three parts:

Soundness If the normalization function is defined, then the output is convertible to the input, and in normal form: $norm(a) = a' \Rightarrow a' : \mathsf{nf} \wedge a \leftrightarrow a'$.

Identification The normalization function yields equal results on convertible terms: $a \leftrightarrow a' \Rightarrow norm(a) = norm(a')$.

Completeness The normalization function will be defined whenever the input term has a normal form: $a \leftrightarrow a' \wedge a' : \mathsf{nf} \Rightarrow norm(a) \neq \bot$.

While identification and completeness are fairly direct consequences of Theorem 1, the proof of soundness requires more work: as explained by Filinski and Rohde, the property is closely related to proofs of adequacy of a denotational semantics with respect to an operational one [11].

$$
\begin{aligned}
\bot \lhd_{obj} b &\Leftrightarrow \text{true}\\
\mathsf{tm}\ a \lhd_{obj} b &\Leftrightarrow a : \mathsf{at}\ \wedge\ a = b\\
\mathsf{rec}\ r \lhd_{obj} [\ell_i = m_i{}^{i \in [k]}] &\Leftrightarrow \{\ell_i \mid i \in [k]\} = \mathsf{dom}(r)\ \wedge\ \forall(i \in [k])\ r.\ell_i \lhd_{meth} m_i\\
f \lhd_{meth} \varsigma(x)a &\Leftrightarrow \forall(d \in \mathbb{O})\, \forall(b \in \mathit{obj})\ d \lhd_{obj} b\ \Rightarrow\ f(d) \lhd (x := b)(a)\\
d \lhd b &\Leftrightarrow d\ (\lhd_{obj})^{\bot\bot}\ b
\end{aligned}
$$

Fig. 4. Relations $\lhd_{obj}\ \subseteq \mathbb{O} \times \mathit{obj}$, $\lhd_{meth}\ \subseteq \mathbb{M} \times \mathit{meth}$, and $\lhd\ \subseteq \mathbb{O}^{\bot\bot} \times \mathit{obj}$

Relating Denotations to NBE Results For an FM-cppo A and relation $R \subseteq A \times \mathit{obj}$ we let $R^{\bot} \subseteq (A \multimap (\mathit{obj}_{\bot})^{\bot\bot}) \times \mathit{ctxt}$ and $R^{\bot\bot} \subseteq A^{\bot\bot} \times \mathit{obj}$ be the relations, resp., defined by:

$$
\begin{aligned}
\varphi\ R^{\bot}\ C[-] \Leftrightarrow \forall(d \in A)\, \forall(b \in \mathit{obj})\ d\ R\ b\ \wedge\ \varphi(d) \neq \bot\ \Rightarrow&\\
\exists(a : \mathsf{nf})\ \varphi(d) = \mathbf{return}(a)\ \wedge\ C[b] \leftrightarrow a&
\end{aligned}
$$

$$
\begin{aligned}
d\ R^{\bot\bot}\ b \Leftrightarrow \forall(\varphi \in (A \multimap \mathit{obj}_{\bot}^{\bot\bot}))\forall(C[-] \in \mathit{ctxt})\ \varphi\ R^{\bot}\ C[-]\ \wedge\ \varphi^*(d) \neq \bot\ \Rightarrow&\\
\exists(a : \mathsf{nf})\ \varphi^*(d) = \mathbf{return}(a)\ \wedge\ C[b] \leftrightarrow a&
\end{aligned}
$$

Using this notation, Figure 4 defines a (recursive) relation $\lhd = (\lhd_{obj})^{\bot\bot} \subseteq \mathbb{O}^{\bot\bot} \times \mathit{obj}$. Note that the existence of such a relation $\lhd$ is not immediately obvious, due to both positive and negative occurrences of the relation in the clause for $\lhd_{meth}$. The existence proof follows the method of Pitts [15] (exploiting the minimal invariant property of the domain $\mathbb{O}$), and is given as Theorem A4 in the appendix.

Lemma 2 (Lifting). *If $d \lhd_{obj} b$ then* $\mathbf{return}(d) \lhd b$.

Proof. By definition, since $\varphi^*(\mathbf{return}\ d) = \varphi(d)$. □

Lemma 3 (Fundamental property of $\lhd$). *Suppose η is an environment and θ is a substitution such that $\eta(x) \lhd_{obj} \theta(x)$ for all $x \in \mathit{fv}(a)$. Then*

1. *for all $a \in \mathit{obj}$, $[\![a]\!]_{\eta} \lhd \theta(a)$, and*
2. *for all $m \in \mathit{meth}$ and $g \in \mathbb{M}$, if $[\![m]\!]_{\eta} = \mathbf{return}\, g$ then $g \lhd_{meth} \theta(m)$.*

Proof. Simultaneously by α-structural induction on a and m, respectively. The cases where a is x or $[\ell_i = m_i{}^{i \in [k]}]$ are easy. If a is $a'.\ell$ then some desugaring of the **match** expression yields $[\![a]\!]_{\eta} = \mathbf{let}\ v' \Leftarrow [\![a']\!]_{\eta}\ \mathbf{in}\ (f_1 \oplus f_2)(v')$ where $f_1 \in (\mathit{obj}_{\bot} \multimap \mathbb{O}^{\bot\bot})$ and $f_2 \in (\mathsf{Rec}_{\mathcal{L}}(\mathbb{M}) \multimap \mathbb{O}^{\bot\bot})$ are

$$f_1(b) = \uparrow(\mathsf{sel}(b, \ell)) \qquad\qquad f_2(r) = r.\ell(\mathsf{rec}\ r)$$

We must prove that $[\![a]\!]_{\eta} \lhd \theta(a)$, i.e., that there exists $b : \mathsf{nf}$ such that both $\mathbf{let}\ v \Leftarrow [\![a]\!]_{\eta}\ \mathbf{in}\ \varphi(v) = \mathbf{return}(b)$ and $C[\theta(a)] \leftrightarrow b$ hold whenever $\varphi\ (\lhd_{obj})^{\bot}\ C[-]$ and $\mathbf{let}\ v \Leftarrow [\![a]\!]_{\eta}\ \mathbf{in}\ \varphi(v) \neq \bot$.

So let $\varphi \ (\lhd_{obj})^{\perp} \ C[-]$ and suppose $\textbf{let } v \Leftarrow [\![a]\!]_{\eta} \textbf{ in } \varphi(v) \neq \perp$. By part 1 of the induction hypothesis, $[\![a']\!]_{\eta} \lhd \theta(a')$, hence for all $\varphi' \ (\lhd_{obj})^{\perp} \ C'[-]$,

$$\begin{aligned}&\textbf{let } v' \Leftarrow [\![a']\!]_{\eta} \textbf{ in } \varphi'(v') \neq \perp \ \Rightarrow \\ &\qquad \exists(b' : \mathsf{nf}) \ \textbf{let } v' \Leftarrow [\![a']\!]_{\eta} \textbf{ in } \varphi'(v') = \textbf{return}(b') \ \wedge \ C'[\theta(a')] \leftrightarrow b'. \qquad (6)\end{aligned}$$

Thus, instantiating (6) with $\varphi'(v') = \textbf{let } v \Leftarrow (f_1 \oplus f_2)(v) \textbf{ in } \varphi(v')$ and $C'[-] = C[[-].\ell]$, and observing that by associativity

$$\textbf{let } v' \Leftarrow [\![a']\!]_{\eta} \textbf{ in } \varphi'(v') \ = \ \textbf{let } v \Leftarrow [\![a]\!]_{\eta} \textbf{ in } \varphi(v) \ \neq \ \perp,$$

we find that $\varphi' \ (\lhd_{obj})^{\perp} \ C'[-]$ implies

$$\exists(b : \mathsf{nf}) \ \textbf{let } v \Leftarrow [\![a]\!]_{\eta} \textbf{ in } \varphi(v) = \textbf{return}(b) \ \wedge \ b \leftrightarrow C'[\theta(a')] = C[\theta(a)]$$

from which we may conclude existence of the required normal form b once we have established $\varphi' \ (\lhd_{obj})^{\perp} \ C'[-]$.

To that end, let $d_0 \lhd_{obj} b_0$ and assume $\varphi'(d_0) \neq \perp$. In particular, $d_0 \neq \perp$, so either $d_0 = \mathsf{tm} \ a_0$ for $a_0 \in obj$, or $d_0 = \mathsf{rec} \ r$ where $\perp \neq r \in \mathsf{Rec}_{\mathcal{L}}(\mathbb{M})$. In the first case, $a_0 : \mathsf{at}$ and $a_0 \leftrightarrow b_0$ by definition of $\lhd_{obj}$. Hence, $\mathsf{tm}(a_0.\ell) \lhd_{obj} b_0.\ell$, and

$$\perp \ \neq \ \varphi'(d_0) \ = \ \textbf{let } v \Leftarrow f_1(a_0) \textbf{ in } \varphi(v) \ = \ \varphi(\mathsf{tm} \ (a_0.\ell))$$

combined with $\varphi \ (\lhd_{obj})^{\perp} \ C[-]$ guarantees the existence of $b' : \mathsf{nf}$ such that $\varphi'(d_0) = \textbf{return}(b)$ and $b' \leftrightarrow C[a_0.\ell] \leftrightarrow C[b_0.\ell] \leftrightarrow C'[b_0]$. In the second case, where $d_0 = \mathsf{rec} \ r$, by the definition of $\lhd_{obj}$ b_0 must be $[\ell_i = m_i{}^{i \in [k]}]$ where $\{\ell_i \mid i \in [k]\} = \mathsf{dom}(r)$ and $r.\ell_i \lhd_{meth} m_i$ for all i. From $\perp \neq \varphi'(d_0)$ we obtain

$$\varphi'(d_0) \ = \ \textbf{let } v \Leftarrow f_2(r) \textbf{ in } \varphi(v) \ = \ \textbf{let } v \Leftarrow r.\ell(\mathsf{rec} \ r) \textbf{ in } \varphi(v).$$

Writing m_i as $\varsigma(x_i)a_i$, the assumption $d_0 \lhd_{obj} b_0$ immediately yields $r.\ell(\mathsf{rec} \ r) \lhd (x_i \mapsto b_0)(a_i)$, so that from the assumption $\varphi \ (\lhd_{obj})^{\perp} \ C[-]$ we obtain $\varphi'(d_0) = \textbf{return}(b')$ and $b' \leftrightarrow C[(x_i \mapsto b_0)(a_i)]]$ for some $b' : \mathsf{nf}$. Thus also $C'[b_0] = C[b_0.\ell_i] \leftrightarrow C[(x_i \mapsto b_0)(a_i)] \leftrightarrow b'$. This proves $\varphi' \ (\lhd_{obj})^{\perp} \ C'[-]$ and concludes this case of the inductive proof.

The case where a is $a'.\ell := m$ is similar, using $\varphi'(v') = \textbf{let } v \Leftarrow (f_1 \oplus f_2)(v) \textbf{ in } \varphi(v)$ where $f_1(b) = \textbf{let } m' \Leftarrow \downarrow_m{}^* [\![m]\!]_{\eta} \textbf{ in } \uparrow(\mathsf{upd}(b, \ell, m'))$ and $f_2(r) = \textbf{let } f \Leftarrow [\![m]\!]_{\eta} \textbf{ in return}(\mathsf{rec} \ r.\ell := f)$, and the context $C'[-] = C[[-].\ell := \theta(m)]$.

Finally, let us consider the case of methods, where m is $\varsigma(x)a$ and we may assume that $x \notin supp(\theta) = \bigcup_{x \in \mathsf{dom}(\theta)}\{supp(\theta(x))\} \cup \mathsf{dom}(\theta)$. Then $[\![m]\!]_{\eta}$ denotes $\textbf{return}(g)$ for $g(d) = \textbf{if } d = \perp \textbf{ then } \perp \textbf{ else } [\![a]\!]_{\eta[x:=d]}$, and $\theta(m) = \varsigma(x)\theta(a)$. We must prove that $[\![m]\!]_{\eta} \lhd_{meth} \theta(m)$, so let $d \lhd_{obj} b$. Observe that for $\eta' = \eta[x := d]$ and $\theta' = (x \mapsto b) \circ \theta$ we have $\eta'(y) \lhd_{obj} \theta'(y)$ for all $y \in Var$, hence by part 1 of the induction hypothesis

$$([\![m]\!]_{\eta})(d) \ = \ [\![a]\!]_{\eta'} \ \lhd \ \theta'(a) \ = \ (x \mapsto b)(\theta(a)).$$

Since this is true for all $d \lhd_{obj} b$ we have proved $[\![m]\!]_{\eta} \lhd_{meth} \theta(m)$. $\square$

Lemma 4 (Reification of related elements).

1. *For all $a \in obj$, if $a : \mathsf{at}$ then $\mathsf{tm}(a) \lhd_{obj} a$.*
2. *For all $d \in \mathbb{O}$, $b \in obj$, if $d \lhd_{obj} b$ and $\downarrow(d) \neq \perp$ then $\downarrow(d) = \mathbf{return}(a)$ for some $a : \mathsf{nf}$ such that $a \leftrightarrow b$.*
3. *For all $f \in \mathbb{M}$, $m \in meth$, if $f \lhd_{meth} m$ and $\downarrow_m(f) \neq \perp$ then $\downarrow_m(f) = \mathbf{return}(m')$ for some $m' : \mathsf{mnf}$ such that $m' \leftrightarrow m$.*

Proof (sketch). Part 1 is immediate from the definition of $\lhd_{obj}$. The proof of the second and third part is by fixed point induction with respect to the admissible predicates $P \subseteq \mathbb{O} \multimap (obj_\perp)^{\perp\perp}$ and $Q \subseteq \mathbb{M} \multimap (meth_\perp)^{\perp\perp}$,

$$\begin{aligned}&P = \{\varphi \mid \varphi\ (\lhd_{obj})^{\perp}\ [-]\}\\ &Q = \{\psi \mid \forall f \lhd_{meth} m.\, \psi(f) \neq \perp\ \Rightarrow\ \exists(m' : \mathsf{mnf})\ \psi(f) = \mathbf{return}(m') \wedge m' \leftrightarrow m\}\end{aligned}$$

More precisely, defining $\Phi : (\mathbb{M} \multimap (meth_\perp)^{\perp\perp}) \to (\mathbb{O} \multimap (obj_\perp)^{\perp\perp})$ by

$$\begin{aligned}\Phi(\psi)(d) = {}&\mathbf{match}\ d\ \mathbf{with}\\ &\quad tm(a) \Rightarrow \mathbf{return}(a)\\ &\quad \mid rec(r) \Rightarrow \mathbf{let}\ (m_\ell \Leftarrow \psi(r.\ell) \mid \ell{\in}\mathsf{dom}(r))\ \mathbf{in}\ \mathbf{return}(\mathsf{obj}\ \{\![\ell = m_\ell{}^{\ell\in\mathsf{dom}(r)}]\!\})\end{aligned}$$

and $\Psi : (\mathbb{O} \multimap (obj_\perp)^{\perp\perp}) \to (\mathbb{M} \multimap (meth_\perp)^{\perp\perp})$ by

$$\Psi(\varphi)(f) = \mathbf{let}\ x \Leftarrow fresh\ \mathbf{in}\ \mathbf{let}\ a \Leftarrow \varphi^*(f^*(\uparrow(\mathsf{var}\, x)))\ \mathbf{in}\ \mathbf{return}(\mathsf{meth}(x, a))$$

we prove $\varphi \in P\ \Rightarrow\ \Psi(\varphi) \in Q$ and $\psi \in Q\ \Rightarrow\ \Phi(\psi) \in P$, for then the definition of $\downarrow\ = lfp(\Phi \circ \Psi) \in P$ and $\downarrow_m = \Phi_m(\downarrow) = lfp(\Psi \circ \Phi) \in Q$ as least fixed points have the required properties, by definition of $(\lhd_{obj})^{\perp}$ and Q, respectively. $\square$

Lemma 5 (Definedness of normal forms). *Suppose that for all $x \in fv(a) \cup fv(m)$, $\eta(x) = \mathsf{tm}(b)$ for some $b \in obj$. Then*

1. *if $a : \mathsf{at}$ then $[\![a]\!]_\eta = \mathbf{return}(\mathsf{tm}\ a')$ for some $a' \in obj$;*
2. *if $a : \mathsf{nf}$ then $\downarrow^*([\![a]\!]_\eta) = \mathbf{return}(a')$ for some $a' \in obj$; and*
3. *if $m : \mathsf{mnf}$ then $\downarrow_m{}^*([\![m]\!]_\eta) = \mathbf{return}(m')$ for some $m' \in meth$.*

Proof. By induction on the derivation of $a : \mathsf{at}$, $a : \mathsf{nf}$ and $m : \mathsf{mnf}$, resp. $\square$

We can now prove correctness: since $\mathsf{tm}(\mathsf{var}\ x) \lhd_{obj} x$ by Lemma 4(1) we have $[\![a]\!]_{\eta_0} \lhd a$ for $\eta_0 = \lambda(x \in Var).\, \mathsf{tm}(\mathsf{var}\ x)$, by Lemma 3. Hence, if $norm(a) \neq \perp$ then $\downarrow([\![a]\!]\,\eta_0) = \mathbf{return}(a')$ for some $a' : \mathsf{nf}$ such that $a \leftrightarrow a'$, by Lemma 4(2), and $norm(a) = \mathbf{return}(a')$.

Conversely, if $a \leftrightarrow a'$ then $\downarrow^*([\![a]\!]_{\eta_0}) = \downarrow^*([\![a']\!]_{\eta_0})$ by Theorem 1. In particular, if $a' : \mathsf{nf}$ then $\downarrow^*([\![a]\!]_{\eta_0}) \neq \perp$ by Lemma 5(2), and we have shown:

Theorem 2 (Correctness).

1. *If $norm(a) \neq \perp$ then $norm(a) = \mathbf{return}(a')$ for some $a' : \mathsf{nf}$ such that $a' \leftrightarrow a$.*
2. *If $a \leftrightarrow a'$ then $norm(a) = norm(a')$.*
3. *If $a \leftrightarrow a'$ for some $a' : \mathsf{nf}$, then $norm(a) \neq \perp$.*

6 Conclusion

We have proved correctness of NBE for Abadi and Cardelli's (untyped) object calculus, giving a semi-decision procedure for the simplest of the equational theories presented in [1]. Shinwell and Pitts prove that the continuation semantics forms an adequate model of the Fresh O'Caml dialect of SML [19]. In this sense, the normalization result leads to an implementation that is correct by construction.

The previous approach of Filinski and Rohde, using a de Bruijn-level naming scheme for the computed normal forms, clearly carries over to object calculus. Correspondingly, our main contribution here is not so much the consideration of the object calculus but working out the details of the construction in the world of nominal sets. We believe that our work provides further evidence to support the point made in [16]: an approach to NBE using nominal sets in the formal development "*allows us to retain the essential simplicity of an informal account [...]*", without obscuring the basic ideas by issues of name generation.

However, while this is true for the *statement* of the properties, the use of the continuation monad certainly complicates some of the *proofs*, as a comparison to [11] shows: Filinski and Rohde prove correctness of NBE for the untyped lambda calculus using standard domain-theoretic methods, and handle name generation by means of 'wrapper functions' and Kripke relations to keep track of used names. The proof of our logical relations lemma (Lemma 3) is less straightforward than that of the corresponding property in [11]. Moreover, their constructions can be implemented in 'conventional' functional programming languages, not relying on language support for freshness. On the other hand, as observed by one of the reviewers, these facts also indicate that NBE may simply not be a good application for demonstrating the rather more powerful machinery of nominal sets: deconstruction and pattern matching of abstract syntax with binders, which is supported by nominal sets, is not necessary for NBE. (Pattern matching will be implicitly used in an implementation, when comparing two normal forms in $(obj_\perp)^{\perp\perp}$ for equality; due to the 'extraction' of a term b from the continuation semantics in (5), the given definition of *norm* is computationally not meaningful.)

One remaining question is how to capture the more refined equational theories of objects presented in [1] which rely on types. The problem here is that *subtyping* is an obstacle to defining a reification map. Another question that we leave open is the generalization from computing normal forms to computing Böhm trees [5], which have a natural analogue in the object calculus. For untyped lambda terms, [11] shows that the domain-theoretic normalization result extends to this infinitary case. It should be interesting to see if a similar generalization is possible within FM-domain theory: for instance, the domain of (lazy) lambda trees used in [11] differs from a correspondingly constructed FM-cpo, in that the FM-cpo cannot contain trees with infinitely many free variables (due to the finite support property).

Acknowledgments I would like thank the reviewers for their valuable comments that helped to improve correctness and readability of the paper.

References

1. M. Abadi and L. Cardelli. *A Theory of Objects.* Springer, 1996.
2. K. Aehlig and F. Joachimski. Operational aspects of untyped normalization by evaluation. *Mathematical Structures in Computer Science*, 14(4):587–611, 2004.
3. T. Altenkirch, M. Hofmann, and T. Streicher. Reduction-free normalisation for a polymorphic system. In *Proc. LICS'96*, pages 98–106, 1996.
4. V. Balat, R. Di Cosmo, and M. P. Fiore. Extensional normalisation and type-directed partial evaluation for typed lambda calculus with sums. In *Proc. POPL'04*, pages 64–76. ACM Press, 2004.
5. H. P. Barendregt. *The Lambda Calculus.* North Holland, 1984.
6. N. Benton and B. Leperchey. Relational reasoning in a nominal semantics for storage. In *Proc. TLCA'05*, volume 3461 of *LNCS*, pages 86–101. Springer, 2005.
7. U. Berger and H. Schwichtenberg. An inverse of the evaluation functional for typed λ-calculus. In *Proc. LICS'91*, pages 203–211. IEEE Computer Society Press, 1991.
8. T. Coquand and P. Dybjer. Intuitionistic model constructions and normalization proofs. *Mathematical Structures in Computer Science*, 7(1):75–94, 1997.
9. U. de'Liguoro. Characterizing convergent terms in object calculi via intersection types. In *Proc. TLCA'01*, pages 315–328, 2001.
10. P. Dybjer and A. Filinski. Normalization and partial evaluation. In *Applied Semantics (APPSEM'00)*, volume 2395 of *LNCS*, pages 137–192. Springer, 2000.
11. A. Filinski and H. K. Rohde. Denotational aspects of untyped normalization by evaluation. *Theoretical Informatics and Applications*, 39(3):423–453, 2005.
12. K. Fisher and J. C. Mitchell. A delegation-based object calculus with subtyping. In *FCT'95*, volume 965 of *LNCS*, pages 42–61. Springer, 1995.
13. Fresh O'Caml. Website at http://www.fresh-ocaml.org/, 2007.
14. S. Lindley and I. Stark. Reducibility and $\top\top$-lifting for computation types. In *Proc. TLCA'05*, volume 3461 of *LNCS*, pages 262–277. Springer, 2005.
15. A. M. Pitts. Relational properties of domains. *Information and Computation*, 127:66–90, 1996.
16. A. M. Pitts. Alpha-structural recursion and induction. *Journal of the ACM*, 53:459–506, 2006.
17. B. Reus and T. Streicher. Semantics and logic of object calculi. *Theoretical Computer Science*, 316:191–213, 2004.
18. M. R. Shinwell. *The Fresh Approach: functional programming with names and binders.* PhD thesis, University of Cambridge Computer Laboratory, Feb. 2005.
19. M. R. Shinwell and A. M. Pitts. On a monadic semantics for freshness. *Theoretical Computer Science*, 342:28–55, 2005.

A Existence Proof

Let *Rel* be the set of finitely supported relations $R \subseteq \mathbb{O} \times obj$ such that $\{d \mid d \; R \; b\}$ forms an admissible subset of $\mathbb{O}$, for all $b \in obj$.

Lemma A1 (Admissibility preservation) *If $R \in Rel$ then $\{d \mid d \; R^{\perp\perp} \; b\}$ is an admissible subset of $\mathbb{O}^{\perp\perp}$, for all $b \in obj$.*

Proof. That $\perp R^{\perp\perp} b$ is immediate from the definition. So let $b \in obj$, $D \subseteq \mathbb{O}^{\perp\perp}$ be a finitely supported directed set such that $d\ R^{\perp\perp}\ b$ holds for all $d \in D$, and let $e = \sqcup D$. To prove $e\ R^{\perp\perp}\ b$ let $\varphi\ R^{\perp}\ C[-]$ and suppose

$$\perp \neq \varphi^*(e) = \sqcup\{\varphi^*(d) \mid d \in D\}.$$

Thus there exists some $d \in D$ such that $\varphi^*(d) \neq \perp$, and for all such d we have that there is $a : \mathsf{nf}$ where $\varphi^*(d) = \textbf{return}(a)$ and $C[b] \leftrightarrow a$ by $d\ R^{\perp\perp}\ b$. (From the discrete ordering on obj it follows that this is the same a for all such d). Thus $\varphi^*(e) = \textbf{return}(a)$ and the result follows. $\square$

For $R, S \in Rel$ let $\Psi_M(R, S) \subseteq \mathbb{M} \times meth$ be

$$\Psi_M(R,S) = \{(f, \varsigma(x)a) \mid \forall(d \in \mathbb{O})\forall(b \in obj)\ d\ R\ b\ \Rightarrow\ f(d)\ S^{\perp\perp}\ (x \mapsto b)(a)\}$$

and $\Psi(R, S) \subseteq \mathbb{O} \times obj$ be such that $(d, b) \in \Psi(R, S)$ holds if and only if

1. $d = \perp$, or
2. $d = \mathsf{tm}(b)$ and $b : \mathsf{at}$, or
3. $d = \mathsf{rec}(r)$, $b = [\ell_i = m_i{}^{i\in[k]}]$ where $\mathsf{dom}(r) = \{\ell_i \mid i \in [k]\}$, and $(r.\ell_i, m_i) \in \Psi_M(R, S)$ for all $i \in [k]$.

Lemma A2 (Admissibility) *For all $R, S \in Rel$, $\Psi(R, S) \in Rel$.*

Proof. This follows easily with Lemma A1. $\square$

We note that Rel is a FM-complete lattice with respect to set inclusion, with meets of finitely supported sets given by set-theoretic intersection. Moreover, by Lemma A2, the *symmetrization* $\Psi^{\S}$ of Ψ,

$$R, S\ \mapsto\ \Psi^{\S}(R, S) = (\Psi(S, R), \Psi(R, S))$$

is a monotone map on $Rel^{op} \times Rel$ which has a least (pre-)fixed point (Δ^-, Δ^+) by a variant of the Knaster-Tarski Fixed Point Theorem [18].

Since then also (Δ^+, Δ^-) is a fixed point, one has $\Delta^+ \subseteq \Delta^-$. For the converse, define for $e \in (\mathbb{O} \multimap \mathbb{O})$ and $R, S \in Rel$,

$$e : R \subset S\ \Leftrightarrow\ \forall(d \in \mathbb{O})\forall(b \in obj).\ (d, b) \in R \Rightarrow (e(d), b) \in S,$$

intuitively stating that e maps R-related elements to S-related elements. A consequence of this definition is the following property.

Lemma A3 *Suppose $e : R \subset S$. Then*

1. $(\varphi, C[-]) \in S^{\perp}\ \Rightarrow\ (\varphi \circ e, C[-]) \in R^{\perp}$
2. $(d, b) \in R^{\perp\perp}\ \Rightarrow\ (\lambda(h \in S^{\perp}).\, d(h \circ e), b) \in S^{\perp\perp}$

for all $\varphi \in (\mathbb{O} \multimap (obj_\perp)^{\perp\perp})$, $C[-] \in ctxt$, $d \in \mathbb{O}^{\perp\perp}$ and $b \in obj$.

Proof. Part (1) is verified straightforwardly using the definition of $R^{\perp}$ and $S^{\perp}$; part (2) then follows from (1) and the definitions of $R^{\perp\perp}$ and $S^{\perp\perp}$, resp. $\square$

Now to show $\Delta^- = \Delta^+$ it suffices to prove that $id : \Delta^- \subset \Delta^+$. Since by the minimal invariant property of $\mathbb{O}$ one has $id = lfp(\delta)$ this follows by a fixed point induction with respect to the admissible (because $\Delta^+ \in Rel$) predicate

$$[\Delta^-, \Delta^+] = \{e \in (\mathbb{O} \multimap \mathbb{O}) \mid e : \Delta^- \subset \Delta^+\}.$$

Clearly $\perp \in [\Delta^-, \Delta^+]$. Moreover, to show $e \in [\Delta^-, \Delta^+] \Rightarrow \delta(e) \in [\Delta^-, \Delta^+]$ it suffices to prove that Ψ satisfies, for all $R, S \in Rel$,

$$e : R \subset S \;\Rightarrow\; \delta(e) : \Psi(S, R) \subset \Psi(R, S), \tag{7}$$

for then $e \in [\Delta^-, \Delta^+]$ yields $\delta(e) : \Psi(\Delta^+, \Delta^-) \subset \Psi(\Delta^-, \Delta^+)$. But the latter is just $\delta(e) : \Delta^- \subset \Delta^+$ by choice of $(\Delta^-, \Delta^+) = \Psi^{\S}(\Delta^-, \Delta^+) \in Rel^{op} \times Rel$, establishing that $e \in [\Delta^-, \Delta^+]$ implies $\delta(e) \in [\Delta^-, \Delta^+]$.

It remains to prove (7). To this end, assume $e : R \subset S$, and let $(d, b) \in \Psi(S, R)$; we show $(\delta(e)(d), b) \in \Psi(R, S)$. This is clear if $\delta(e)(d) = \perp$, so assume $\delta(e)(d) \neq \perp$. Then, by definition of δ, either $d = \mathsf{tm}(a)$ for some $a \in obj$ and $\delta(e)(d) = \mathsf{tm}(a)$, or else $d = \mathsf{rec}(r)$ for some $\perp \neq r \in \mathsf{Rec}_{\mathcal{L}}(\mathbb{M})$ and $\delta(e)(r).\ell = \delta_M(e)(r.\ell)$ for all $\ell \in \mathsf{dom}(r)$. In the former case, $(\delta(e)(d), b) \in \Psi(R, S)$ follows directly from the assumption $(d, b) \in \Psi(S, R)$, so let us consider the other case.

The assumption $(\mathsf{rec}(r), b) \in \Psi(S, R)$ yields that b is of the form $[\ell_i = m_i{}^{i \in [k]}]$ where $\mathsf{dom}(r) = \{\ell_i \mid i \in [k]\}$, and it remains to show $(\delta_M(e)(r.\ell_i), m_i) \in \Psi_M(R, S)$ for all $i \in [k]$. By the precondition in (7), $(e(d'), b') \in S$ for all $(d', b') \in R$. Supposing that $m_i = \varsigma(x_i)a_i$ then, since $(r.\ell_i, m_i) \in \Psi_M(S, R)$, we have

$$\forall(d' \in \mathbb{O})\forall(b' \in obj).\ (d', b') \in R \;\Rightarrow\; (r.\ell_i(e(d')), (x_i \mapsto b')(a_i)) \in R^{\perp\perp}. \tag{8}$$

Using $e : R \subset S$ we may instantiate Lemma A3(2) by the right-hand side of (8) to obtain for all $d' \in \mathbb{O}$ and all $b' \in obj$:

$$(d', b') \in R \;\Rightarrow\; (\lambda(h \in S^{\perp}).\, r.\ell_i(e(d'))(h \circ e), (x_i \mapsto b')(a_i)) \in S^{\perp\perp},$$

which shows the required property $(\delta_M(e)(r.\ell_i), m_i) \in \Psi_M(R, S)$.

Hence we have shown (7) and may define $\lhd_{obj}$ to equal $\Delta^- = \Delta^+$, to obtain the following:

Theorem A4 (Existence) *There exists a relation $\lhd_{obj} \in Rel$ such that $\lhd_{obj} = \Psi(\lhd_{obj}, \lhd_{obj})$.*