Logical Semantics for the First Order ς-Calculus

Steffen van Bakel[1]* and Ugo de'Liguoro[2]**

[1] Department of Computing, Imperial College,
180 Queen's Gate, London SW7 2BZ, UK,
svb@doc.ic.ac.uk
[2] Dipartimento di Informatica, Università di Torino,
Corso Svizzera 185, 10149 Torino, Italy
deliguoro@di.unito.it

Abstract. We investigate logical semantics of the first order ς-calculus. An assignment system of predicates to first order typed terms of the OB_1 calculus is introduced. We define retraction models for that calculus and an interpretation of terms, types and predicates into such models. The assignment system is then proved to be sound and complete w.r.t. retraction models.

1 Introduction

The essence of logical semantics of a calculus is a system of predicates and a relation of satisfiability, such that the meaning of a term in the calculus can be identified with the set of predicates it satisfies. Examples are intersection types for the type-free λ-calculus [7, 6], pre-locales for typed λ-calculi and domain logic [4], Hennessy-Milner logic for CCS terms [15, 3]. The present work is aimed at defining a logical semantics suitable for typed object calculi.

In [12] it is shown that certain characterizations of reduction properties of pure λ-terms via intersection types (for which see e.g. [10, 16, 6, 11]) are smoothly inherited by the type-free ς-calculus, provided we extend the intersection type discipline to a λ-calculus with records, and interpret ς-terms using the self-application interpretation of [14]. Nonetheless the focus of research in the area of object calculi is on typed systems and typed equational theories. To make our approach applicable to the latter case we have to put on a clear footing the idea of an assignment system of predicates to typed objects: a first investigation is [13].

For monomorphic typed calculi predicates differ from types in that each term has exactly one type, but satisfies (often infinitely) many predicates. This difference is blurred in the polymorphic case (indeed, in the literature, Curry types, intersection types and ML types are considered as forms of polymorphism), but it is still true that, while predicates give partial information about the behavior of single terms, types are concerned with general properties of the system, like strong normalization for typed λ-calculi, or error-freeness of the reducts of typed terms in object calculi.

* Partially supported by EU project IST-1999-20975, SecSafe
** Partially supported by MURST Cofin'01 COMETA Project, IST-2001-33477 DART Project and IST-2001-322222 MIKADO Project. The funding bodies are not responsible for any use that might be made of the results presented here.

The solution we propose is to consider types as languages of predicates, or even better as theories. The denotation of a term is then a set of predicates closed under conjunction and logical implication (technically a filter), but when such a denotation is relativized to a type, which is the counterpart of typing the term, its denotation is restricted to the language associated with that type. This suggests a natural interpretation of features of polymorphic typed systems, as it is the case of subtyping: $A <: B$ if the theory associated to B is "included" into the theory of A, which means that its discriminating power is at most that of A (for a topological interpretation of the same idea, and its relation to realizability models and PER inclusion see [13]).

In the present paper we investigate logical semantics of the first order ς-calculus of [1], called there system OB_1. This is the core of the object calculi studied in that book, even if it is poorly expressive and does not include any form of subtyping. Still it is an interesting case study, as the recursive nature of types is challenging to model (it is the most complex and contrived part of the semantic constructions in [2, 1, 8]). It comes out that the filter model of the typed calculus has the structure of a retraction model, in the sense of [18], where retractions map filters of predicates to their intersection with the language associated to the given type. This leads to a completeness theorem of the assignment system with respect to retraction models of the calculus. We stress that languages, which define the retractions over the filter model, are inductively defined sets of predicates: a concept of lower logical complexity, and much easier to understand, than fixed-points of contractive operators over ideals or over complete uniform PERs.

2 Assignment for the typed ς-calculus

In this section we introduce the calculus, its types and typing rules, the syntax of the predicates and an assignment system, to syntactically derive judgements associating predicates to typed terms under the assumption of similar judgements about a finite set of typed variables. Predicates are transparently intersection types for a λ-calculus with records, and come from [12]. The essential difference is that the set of predicates is stratified into languages, in such a way that whenever a predicate can be deduced for a term a^A, it belongs to the language $\mathcal{L}_A$ associated with A.

2.1 The calculus

Definition 1 (Untyped terms). *Let $L = \{\ell_i \mid i \in N\}$ be a denumerable set of* labels*. The terms of the first order ς-calculus are defined through the following grammar.*

$$a, b ::= x \mid [\ell_i = \varsigma(x_i) b_i{}^{i\in I}] \mid a.\ell \mid a.\ell \Leftarrow \varsigma(x) b$$

In the expression $\varsigma(x^A)b$, the operator $\varsigma(\cdot)$ binds x in b; free and bound variables are defined as usual. Terms are considered equal modulo α-conversion, i.e. up to renaming of bound variables.

Definition 2 (Reduction). *On terms, the* reduction relation *is defined as the contextual, transitive closure of the following reduction rules:*

$$\begin{array}{rcl} [\ell_i = \varsigma(x_i) b_i{}^{i\in I}].\ell_j & \to & b_j\{x_j \leftarrowtail [\ell_i = \varsigma(x_i) b_i{}^{i\in I}]\} \\ [\ell_i = \varsigma(x_i) b_i{}^{i\in I}].\ell_j \Leftarrow \varsigma(x) b & \to & [\ell_i = \varsigma(x_i) b_i{}^{i\in I\setminus j}, \ell_j = \varsigma(x) b] \end{array}$$

where $j \in I$ and $a\{x \leftarrowtail b\}$ is the substitution of x by b in a, avoiding variable clashes.

The reduction relation is confluent (see [1] Ch. 6). Terms do not necessarily have a normal form: e.g. $\Omega \equiv [\ell = \varsigma(x)x.\ell].\ell$ is such that $\Omega \to \Omega$.

2.2 The typed system

The following is a presentation of the system OB_1 of [1], with minor changes consisting in writing a^A instead of $a{:}A$, and omitting rules for deriving well formed types and contexts: first order types are indeed defined by a simple inductive definition.

Definition 3 (Types). *Let $\mathcal{K}$ be a set of type constants, ranged over by K. The set of* types *is defined by the following grammar:*

$$A, B ::= K \mid [\ell_i : B_i \ ^{i \in I}]$$

where I is a finite set of indexes.

In the present setting, a *context* for a type judgement is just a finite set E of type decorated variables, of the shape x^A.

Definition 4. *The* type judgements *are defined by the following natural deduction system (where $A = [\ell_k : B_k \ ^{k \in I}]$):*

$$(\textit{Var})\ \frac{}{E \vdash x^A}\ (x^A \in E) \qquad (\textit{Val Object})\ \frac{E, x_i^A \vdash b_i^{B_i}}{E \vdash [\ell_i = \varsigma(x_i^A)b_i^{B_i\ i \in I}]^A}\ (\forall i \in I)$$

$$(\textit{Val Select})\ \frac{E \vdash a^A}{E \vdash (a^A.\ell_j)^{B_j}}\ (j \in I) \quad (\textit{Val Update})\ \frac{E \vdash a^A \quad E, x^A \vdash b^{B_j}}{E \vdash (a^A.\ell_j \Leftarrow \varsigma(x^A)b^{B_j})^A}\ (j \in I)$$

Having adopted the notation x^A for a term variable x of type A, the context E becomes redundant. We keep it, however, since this turns out to be useful when introducing bases in the subsequent section.

Reduction among typed terms is defined by adapting Definition 2 in the obvious way. The main result about this system (and all its extensions in [1]) is that types are preserved under reduction: since a term of the form $[\ell_i = \varsigma(x^A)b].\ell_j$ or of the form $[\ell_i = \varsigma(x^A)b].\ell_j \Leftarrow \varsigma(y^A)c$ has no type if $i \neq j$, we may conclude that the reduction of typed terms will never get stuck into not well formed terms (see [1] Ch. 7). Typed terms do not necessarily normalize, however: $\vdash \Omega^A \equiv ([\ell = \varsigma(x^{[\ell:A]})x.\ell].\ell)^A$ is derivable in the empty context for all types A.

2.3 A predicate system

Much in the style of [7], in this section we will present a notion of *intersection types*, called *predicates* here; using these, we will define a notion of *predicate assignment*, which will consists basically of associating a predicate to a typed term.

Definition 5 (Predicates). *The set $\mathcal{L}$ of* predicates *is inductively defined by:*

$$\sigma, \tau ::= \kappa \mid \omega \mid (\sigma \to \tau) \mid (\sigma \wedge \tau) \mid \langle \ell_i : \sigma_i{}^{i \in I} \rangle$$

where κ ranges over a countable set of atoms. On predicates a preorder $\leq$ is inductively defined by:

$$\begin{array}{rcl}
\sigma & \leq & \sigma \\
\sigma & \leq & \omega \\
\omega & \leq & \omega \to \omega \\
(\sigma \to \tau) \wedge (\sigma \to \rho) & \leq & \sigma \to (\tau \wedge \rho) \\
\rho \leq \sigma \ \wedge \ \tau \leq \mu & \Rightarrow & \sigma \to \tau \leq \rho \to \mu \\
\sigma \wedge \tau & \leq & \sigma, \ \sigma \wedge \tau \leq \tau \\
\sigma \leq \tau \ \wedge \ \sigma \leq \rho & \Rightarrow & \sigma \leq \tau \wedge \rho \\
\sigma \leq \tau & \Rightarrow & \langle \ell : \sigma \rangle \leq \langle \ell : \tau \rangle \\
\langle \ell_i : \sigma_i{}^{i \in I} \rangle \wedge \langle \ell_j : \tau_j{}^{j \in J} \rangle & \leq & \langle \ell_k : \rho_k{}^{(k \in I \cup J)} \rangle, \text{ where } \begin{cases} \rho_k = \sigma_k \wedge \tau_k, & \text{if } k \in I \cap J, \\ \rho_k = \sigma_k, & \text{if } k \in I \setminus J, \\ \rho_k = \tau_k, & \text{if } k \in J \setminus I \end{cases} \\
\langle \ell_i : \sigma_i{}^{i \in I} \rangle & \leq & \langle \ell_j : \sigma_j{}^{j \in J} \rangle, \text{ if } J \subseteq I \\
\sigma \leq \tau \leq \rho & \Rightarrow & \sigma \leq \rho
\end{array}$$

Finally $\sigma = \tau \iff \sigma \leq \tau \leq \sigma$.

Atomic predicates κ are intended to describe elements of atomic type in the domain of interpretation; $\sigma \to \tau$ is the property of functions sending elements satisfying σ into elements satisfying τ; $\langle \ell_i : \sigma_i{}^{i \in I} \rangle$ is the property of records having values that satisfy σ_i associated with the field ℓ_i for all $i \in I$. Predicates ω and $\sigma \wedge \tau$ mean truth and conjunction respectively; $\sigma \leq \tau$ reads as 'σ implies τ'.

In the following we consider as 'types' also arrow types $A \to B$: functional types are indeed implicit in the interpretation of objects (especially of methods) but do not appear in the syntax of the calculus OB_1 (but they do in the calculi in [1] enriched with lambda abstraction and functional application). Here their use allows for more transparent notations.

Definition 6 (Languages). *The set of all predicates $\mathcal{L}$ is stratified into a family $\{\mathcal{L}_A\}_A$ of sets of predicates called* languages*, indexed over types such that:*

1. *any κ belongs exactly to one $\mathcal{L}_K$, for some $K \in \mathcal{K}$;*
2. *any $\mathcal{L}_A$ is the least set (including atoms if $A \equiv K$) such that:*

$$\frac{}{\omega \in \mathcal{L}_A} \qquad \frac{\sigma \in \mathcal{L}_A \quad \tau \in \mathcal{L}_A}{\sigma \wedge \tau \in \mathcal{L}_A} \qquad \frac{\sigma \in \mathcal{L}_A \quad \tau \in \mathcal{L}_B}{\sigma \to \tau \in \mathcal{L}_{A \to B}}$$

$$\frac{\sigma \in \mathcal{L}_{A \to B_j}}{\langle \ell_j : \sigma \rangle \in \mathcal{L}_A} \, (A = [\ell_i : B_i{}^{i \in I}], j \in I) \qquad \frac{\sigma \in \mathcal{L}_A}{\tau \in \mathcal{L}_A} \, (\sigma \leq \tau)$$

A *statement* is an expression of the shape $a^A : \sigma$, where a is a term, A is a type, such that there exists E with $E \vdash a^A$, and σ is a predicate, and a is called the *subject* of this statement.

A *basis* Γ is a finite set of statements with only (distinct) term variables as subject, of which the predicate is not ω. We say that Γ *preserves languages* if $\sigma \in \mathcal{L}_A$ whenever $x^A : \sigma \in \Gamma$.

If E is a context and Γ a basis, we say that E *fits into* Γ, written $E \triangleleft \Gamma$, if $x^A : \sigma \in \Gamma$ implies $x^A \in E$. We say that two bases Γ_0, Γ_1 are *compatible* if there exists a context E including all variables occurring in both Γ_0 and Γ_1, fitting into both of them.

Definition 7 (Predicate Assignment). *Let $A \equiv [\ell_i : B_i{}^{i\in I}]$ and B, B_i be any type, then:*

$$(\textit{Var})\ \frac{}{\Gamma \vdash x^B : \sigma}\ (x^B : \sigma \in \Gamma)$$

$$(\textit{Type Object})\ \frac{\Gamma, x_i^A : \sigma_i \vdash b_i^{B_i} : \tau_i}{\Gamma \vdash [\ell_i = \varsigma(x_i^A) b_i{}^{i\in I}]^A : \langle \ell_j : \sigma_j \to \tau_j{}^{j\in J} \rangle}\ (\forall i \in I \ \wedge\ J \subseteq I)$$

$$(\textit{Val Select})\ \frac{\Gamma \vdash a^A : \langle \ell_j : \sigma_j \to \tau_j{}^{j\in J} \rangle \quad \Gamma \vdash a^A : \sigma_k}{\Gamma \vdash a.\ell_k^{B_k} : \tau_k}\ (k \in J)$$

$$(\textit{Val Update})\ \frac{\Gamma \vdash a^A : \langle \ell_j : \sigma_j{}^{j\in J} \rangle \quad \Gamma, y^A : \sigma \vdash b^{B_k} : \tau}{\Gamma \vdash (a.\ell_k \Leftarrow \varsigma(y^A) b)^A : \langle \ell_j : \sigma_j{}^{j \in J\setminus k}, \ell_k : \sigma \to \tau \rangle}\ (k \in J)$$

plus the following 'logical' rules:

$$(\omega)\ \frac{E \vdash a^B}{\Gamma \vdash a^B : \omega}\ (E \triangleleft \Gamma) \quad (\wedge I)\ \frac{\Gamma \vdash a^B : \sigma \quad \Gamma \vdash a^B : \tau}{\Gamma \vdash a^B : \sigma \wedge \tau} \quad (\leq)\ \frac{\Gamma \vdash a^B : \sigma \quad \sigma \leq \tau}{\Gamma \vdash a^B : \tau}$$

As a straightforward induction shows, if all bases in the derivation of $\Gamma \vdash a^A : \sigma$ preserve languages, then $\sigma \in \mathcal{L}_A$.

We remark that in rule (*Type Object*) it is not required that the σ_i are equal, not even pairwise consistent (but for the fact that they belong to the same language $\mathcal{L}_A$). This should be compared to rule (*Val Update*), which allows for replacing the subexpression σ_k in the predicate $\langle \ell_j : \sigma_j{}^{j\in J} \rangle$ of the first premise by the completely unrelated predicate $\sigma \to \tau$ in the conclusion. This is sound, however, because of rule (*Val Select*), which checks in the crucial place that the antecedent of the arrow holds of a^A, to which the self variable x_k^A is bound.

These features, which surely sound odd to readers familiar with the literature on object calculi, are indeed essential. Suppose in fact that

$$A \equiv [\ell_0 : \mathsf{Int}, \ell_1 : \mathsf{Int}] \text{ and } a \equiv [\ell_0 = \varsigma(x^A)1, \ell_1 = \varsigma(x^A)x.\ell_0]$$

(using a constant 1 of type Int), so that $\vdash a^A$. Then

$$\dfrac{x^A{:}\omega \vdash 1 : \mathsf{Odd} \qquad \dfrac{x^A{:}\langle \ell_0 : \omega \to \mathsf{Odd} \rangle \vdash x^A : \langle \ell_0 : \omega \to \mathsf{Odd} \rangle \quad x^A{:}\langle \ell_0 : \omega \to \mathsf{Odd} \rangle \vdash x^A : \omega}{x^A{:}\langle \ell_0 : \omega \to \mathsf{Odd} \rangle \vdash (x.\ell_0)^{\mathsf{Int}} : \mathsf{Odd}}}{\vdash a^A{:}\langle \ell_0 : \omega \to \mathsf{Odd}, \ell_1 : \langle \ell_0 : \omega \to \mathsf{Odd} \rangle \to \mathsf{Odd} \rangle}$$

where ℓ_0 is a field and ℓ_1 is the method $\texttt{get}\ell_0$. By rule (*Val Update*) one might derive the seemingly incorrect:

$$\frac{\vdash a^A : \langle \ell_0 : \omega \to \mathsf{Odd}, \ell_1 : \langle \ell_0 : \omega \to \mathsf{Odd}\rangle \to \mathsf{Odd}\rangle \quad y^A{:}\omega \vdash 2 : \mathsf{Even}}{\vdash (a.\ell_0 \Leftarrow \varsigma(y^A)2)^A : \langle \ell_0 : \omega \to \mathsf{Even}, \ell_1 : \langle \ell_0 : \omega \to \mathsf{Odd}\rangle \to \mathsf{Odd}\rangle}$$

This makes sense, however, since it simply tells that if the value at ℓ_0 is an odd integer, then the method ℓ_1 will return an odd integer; it also tells that this is vacuously true of the actual object $(a.\ell_0 \Leftarrow \varsigma(y^A)2)^A$, since it has an even integer at ℓ_0. Moreover it is harmless: $(a.\ell_0 \Leftarrow \varsigma(y^A)2).\ell_1 \xrightarrow{*} 2$ and we clearly assume that $\nvdash 2 : \mathsf{Odd}$; nonetheless $\nvdash (a.\ell_0 \Leftarrow \varsigma(y^A)2).\ell_1 : \mathsf{Odd}$, because rule (*Val Select*) does not apply since $\nvdash (a.\ell_0 \Leftarrow \varsigma(y^A)2) : \langle \ell_0 : \omega \to \mathsf{Odd}\rangle$.

On the other hand the following odd-looking assignment is legal as well, this time by rule (*Type Object*):

$$\frac{x^A{:}\langle \ell_0 : \omega \to \mathsf{Even}\rangle \vdash x^A : \langle \ell_0 : \omega \to \mathsf{Even}\rangle \quad x^A{:}\langle \ell_0 : \omega \to \mathsf{Even}\rangle \vdash x^A : \omega}{x^A{:}\omega \vdash 1 : \mathsf{Odd} \qquad x^A{:}\langle \ell_0 : \omega \to \mathsf{Even}\rangle \vdash (x.\ell_0)^{\mathsf{Int}} : \mathsf{Even}}$$
$$\vdash a^A{:}\langle \ell_0 : \omega \to \mathsf{Odd}, \ell_1 : \langle \ell_0 : \omega \to \mathsf{Even}\rangle \to \mathsf{Even}\rangle$$

In the last case, however, the apparently odd predicate we deduce, is of use to conclude by rule (*Val Update*):

$$\frac{\vdash a^A : \langle \ell_0 : \omega \to \mathsf{Odd}, \ell_1 : \langle \ell_0 : \omega \to \mathsf{Even}\rangle \to \mathsf{Even}\rangle \quad y^A : \omega \vdash 2 : \mathsf{Even}}{\vdash (a.\ell_0 \Leftarrow \varsigma(y^A)2)^A : \langle \ell_0 : \omega \to \mathsf{Even}, \ell_1 : \langle \ell_0 : \omega \to \mathsf{Even}\rangle \to \mathsf{Even}\rangle}$$

which is what we expected.

The next lemma will be of use in the last section. Let $\Gamma \leq \Gamma'$ mean that for all $x^A{:}\tau \in \Gamma'$ there exists $\sigma \leq \tau$ such that $x^A{:}\sigma \in \Gamma$.

Lemma 8. *1. $\Gamma \leq \Gamma'$ and $\Gamma' \vdash a^A{:}\sigma$ implies $\Gamma \vdash a^A{:}\sigma$.*
2. If Γ_0, Γ_1 are compatible bases, then there exists the basis $\Gamma_0 \wedge \Gamma_1$ which is the greatest one such that $\Gamma_0 \wedge \Gamma_1 \leq \Gamma_i$ for $i = 0, 1$.

Proof. The first part is proved by induction over the derivation of $\Gamma' \vdash a^A{:}\sigma$, using $(\leq)$. For the second, let $\Gamma_0 \wedge \Gamma_1$ be the basis including exactly the statements $x^A{:}\sigma$ such that either $x^A{:}\sigma$ is in one of the two basis and not in the other, or $x^A{:}\sigma_0 \in \Gamma_0$, $x^A{:}\sigma_1 \in \Gamma_1$ and $\sigma \equiv \sigma_0 \wedge \sigma_1$. ∎

We end this section by stating, without proof, the main theorem about syntactical properties of the assignment system. It establishes that predicates are invariant under conversion.

Theorem 9 (Subject reduction and expansion).

1. If $\Gamma \vdash a^A{:}\rho$, and $a \to a'$, then $\Gamma \vdash a'^A{:}\rho$.
2. If $\Gamma \vdash a^A{:}\rho$ and $a' \to a$ where $E \vdash a'^A$ for $E \triangleleft \Gamma$, then $\Gamma \vdash a'^A{:}\rho$.

3 Models and logical semantics

There is no definite agreement about what should be considered as a model of object calculi. Even [1] does not give a general definition of this concept. Rather it is commonly held, especially after Cardelli's seminal work on records calculi, that it should be a model of the λ-calculus including operators to build, access and modify finite records, often seen as finite functions over a set of labels.

Definition 10. *We call a structure* $\mathcal{D} = \langle D, L, \mathsf{emp}, \mathsf{lcond}, \mathsf{sel}\rangle$ *an* untyped ς-model *if:*

- D *is a* λ*-model;*
- $L = \{\ell_i \mid i \in N\}$ *is a denumerable set of labels;*
- $\mathsf{emp} \in D$*;*
- $\mathsf{sel} : D \times L \to D$*;*
- $\mathsf{lcond} : D \times L \times D \to D$

such that (writing lcond *and* sel *in a Curryfied form):*

1. $\mathsf{sel}(\mathsf{lcond}\, x\, \ell_i\, y)\ell_i = y$,
2. $i \neq j \Rightarrow \mathsf{sel}(\mathsf{lcond}\, x\, \ell_i\, y)\ell_j = \mathsf{sel}\, x\, \ell_j$,
3. $i \neq j \Rightarrow \mathsf{lcond}(\mathsf{lcond}\, x\, \ell_i\, y)\, \ell_j\, z = \mathsf{lcond}(\mathsf{lcond}\, x\, \ell_j\, z)\, \ell_i\, y$.

emp is the empty record; sel is a selection operator, depending on its second argument for the field to be selected on its first argument; lcond is a conditional update operator, setting to the value of its third argument the field of its first argument at the label which is the second argument. Note that, due to the untyped nature of the structure, nothing prevents from field selection or field update of some non record element of the domain.

An untyped ς-model is a particular case of what is called a λ, record-combinatory structure in [17] ch. 10. Differences are that here $\mathcal{D}$ is a λ-model, instead of a partial combinatory algebra, and the third axiom about lcond which is not in the original definition. The present choices allow for a simpler treatment and are satisfied by the untyped structure in [1] ch. 14, which is the only denotational model of the ς-calculus in the literature.

Since any $\mathcal{D}$ is a λ-model, we shall freely use abstraction notation. Moreover, we use the abbreviations:

$$\begin{aligned} \langle\cdot\rangle &= \mathsf{emp} \\ \langle \ell_i = d_i{}^{i\in\{1,\ldots,n\}}\rangle &= \mathsf{lcond}(\ldots(\mathsf{lcond}\, \mathsf{emp}\, \ell_1\, d_1)\ldots)\ell_n\, d_n \\ d\cdot\ell_i &= \mathsf{sel}\, d\, \ell_i \\ d\cdot\ell_i := e &= \mathsf{lcond}\, d\, \ell_i\, e \end{aligned}$$

A structure of this form can be constructed by solving the domain equation:

$$D = At + [L \to D] + [D \to D] \tag{1}$$

where At is a domain interpreting atomic (namely ground) types. This equation appears in [9, 5], and is essentially the same as in [1], where it is used to build a model of the (second order) typed ς-calculus.

Definition 11. *To each predicate σ we associate a subset $[\![\sigma]\!]^{\mathcal{D}}_{\eta} \subseteq D$ (or simply $[\![\sigma]\!]_{\eta}$ when $\mathcal{D}$ is clear from the context), where η sends each predicate atom κ to some subset of D, and $\eta(\kappa) \subseteq [\![K]\!]$ when $\kappa \in \mathcal{L}_K$ for some constant type K:*

1. $[\![\omega]\!]_{\eta} = D$,
2. $[\![\kappa]\!]_{\eta} = \eta(\kappa)$,
3. $[\![\sigma \wedge \tau]\!]_{\eta} = [\![\sigma]\!]_{\eta} \cap [\![\tau]\!]_{\eta}$,
4. $[\![\sigma \to \tau]\!]_{\eta} = \{d \in D \mid \forall e \in [\![\sigma]\!]_{\eta}.\ de \in [\![\tau]\!]_{\eta}\}$,
5. $[\![\langle \ell_i : \sigma_i \,^{i \in I} \rangle]\!]_{\eta} = \{d \in D \mid \forall i \in I.\ d \cdot \ell_i \in [\![\sigma_i]\!]_{\eta}\}$.

The latter definition formalizes the intended meaning of predicates by defining their extensions; the subsequent proposition states that implication corresponds to set theoretic inclusion of predicate denotations as expected.

Proposition 1. *If $\sigma \leq \tau$ then, for any η, $[\![\sigma]\!]_{\eta} \subseteq [\![\tau]\!]_{\eta}$.*

Definition 12. *A* type interpretation *over $\mathcal{D}$ is a mapping associating with each type A a subset $[\![A]\!]^{\mathcal{D}} \subseteq D$. It is said to be* consistent *with the predicate interpretation $[\![\,\cdot\,]\!]_{\eta}$ if $\sigma \in \mathcal{L}_A$ implies $[\![\sigma]\!]_{\eta} \subseteq [\![A]\!]$.*

Previous definitions provide the essentials to give meaning to $a^A : \sigma$ and to judgments $\Gamma \vdash a^A : \sigma$.

Definition 13. *Suppose that $\mathcal{D}$ is an untyped ς-model. Let the type interpretation and the predicate interpretation be consistent, E be a context, Γ a basis and ξ a term environment:*

1. *$\xi \models E$ if $\xi(x^A) \in [\![A]\!]^{\mathcal{D}}$ whenever $x^A \in E$;*
2. *$E \models a^A$ if for all ξ s.t. $\xi \models E$, $[\![a^A]\!]^{\mathcal{D}}_{\xi} \in [\![A]\!]^{\mathcal{D}}$;*
3. *$\xi \models \Gamma$ if $x^A : \sigma \in \Gamma$ implies $\xi(x^A) \in [\![\sigma]\!]_{\eta} \subseteq [\![A]\!]^{\mathcal{D}}$;*
4. *$\Gamma \models a^A : \sigma$ if for all ξ s.t. $\xi \models \Gamma$, $[\![a^A]\!]^{\mathcal{D}}_{\xi} \in [\![\sigma]\!]_{\eta} \subseteq [\![A]\!]^{\mathcal{D}}$.*

3.1 A model of retractions

Let D be any domain solving the equation (1). Following [18], a *retraction* over D is a continuous function $\rho : D \to D$ such that $\rho^2 = \rho \circ \rho = \rho$. Types can be interpreted by means of retractions by setting $[\![A]\!] = \{d \in D \mid \rho_A(d) = d\}$, which is the same as the range of ρ_A. For basic types one may choose $\rho_K(d) = d$ if $d \in At$, else $\bot$.

Proposition 2. *Let $A \equiv [\ell_i : B_i \,^{i \in I}]$: if ρ_{B_i} is a retraction for all $i \in I$, then there exists a retraction ρ_A such that*

$$\rho_A(d) = \langle \ell_i = \rho_{A \to B_i}(d \cdot \ell_i) \,^{i \in I} \rangle,$$

where $\rho_{A \to B}(d) = \lambda x.\rho_B(d(\rho_A(x)))$ (indeed $\rho_{A \to B}$ is a retraction, if ρ_A and ρ_B are).

Proof. The function (in Curryfied form)

$$\Upsilon_A f d = \langle \ell_i = \lambda x.\rho_{B_i}((d \cdot \ell_i)(fx))^{\,i\in I}\rangle$$

is continuous, hence it has a fixed-point $\rho_A = \mathit{Fix}(\Upsilon_A) = \bigsqcup_n \Upsilon_A^{(n)}$, where $\Upsilon^{(0)} = \lambda x.\bot$, $\Upsilon_A^{(n+1)} = \Upsilon_A(\Upsilon_A^{(n)})$. By its definition we have

$$\rho_A(d) = \langle \ell_i = \lambda x.\rho_{B_i}((d \cdot \ell_i)(\rho_A(x)))^{\,i\in I}\rangle = \langle \ell_i = \rho_{A\to B_i}(d \cdot \ell_i)^{\,i\in I}\rangle.$$

Observe that this is indeed a retraction:

$$\rho_A^2(d) = \bigsqcup_n \Upsilon_A^{(n)}(\bigsqcup_m \Upsilon_A^{(m)}(d)) = \bigsqcup_{n,m} \Upsilon_A^{(n)}(\Upsilon_A^{(m)}(d)) = \bigsqcup_{n+m} \Upsilon_A^{(n+m)}(d) = \rho_A(d).$$

■

We say that $(\mathcal{D}, \{\rho_A\}_A)$ is a *retraction model* if $\mathcal{D}$ is an untyped ς-model and $\{\rho_A\}_A$ is a family of retractions such that $\rho_A(d) = \langle \ell_i = \rho_{A\to B_i}(d \cdot \ell_i)^{\,i\in I}\rangle$, where $A \equiv [\ell_i : B_i{}^{\,i\in I}]$.

Definition 14. *Let $(\mathcal{D}, \{\rho_A\}_A)$ be a retraction model. The* typed interpretation $[\![a^A]\!]_\xi^{\mathcal{D}}$, *where ξ is an environment associating with each term variable an element of D, is inductively defined by:*

$$\begin{array}{rl}
[\![x^A]\!]_\xi & = \xi(x) \\
[\![[\ell_i = \varsigma(x_i^A) b_i^{B_i}{}^{\,i\in I}]]\!]_\xi & = \langle \ell_i = \lambda d.[\![b_i^{B_i}]\!]_{\xi[x_i:=\rho_A(d)]}{}^{\,i\in I}\rangle \\
[\![(a^A.\ell_i)^{B_i}]\!]_\xi & = ([\![a^A]\!]_\xi \cdot \ell_i)[\![a^A]\!]_\xi \\
[\![a^A.\ell_i \Leftarrow \varsigma(x^A) b^{B_i}]\!]_\xi & = [\![a^A]\!]_\xi \cdot \ell_i := \lambda d.[\![b^{B_i}]\!]_{\xi[x:=\rho_A(d)]}.
\end{array}$$

Theorem 15 (Soundness of the type system w.r.t. retraction models).
If $E \vdash a^A$ then $E \models a^A$.

Proof. By induction over the derivation of $E \vdash a^A$ we prove that $\rho_A([\![a^A]\!]_\xi) = [\![a^A]\!]_\xi$ for any environment ξ such that $\xi \models E$. ■

Lemma 16. *Suppose that the image of $\eta(\kappa)$ under ρ_K is included into $\eta(\kappa)$ when $\kappa \in \mathcal{L}_K$. If $\sigma \in \mathcal{L}_A$ and $d \in [\![\sigma]\!]_\eta$ then $\rho_A(d) \in [\![\sigma]\!]_\eta$.*

Proof. By induction on σ. Cases ω and κ are trivial, by definition and hypothesis respectively. Case $\sigma \wedge \tau \in \mathcal{L}_A$ is immediate by induction, since then $\sigma, \tau \in \mathcal{L}_A$.

Case $\sigma \to \tau \in \mathcal{L}_{A\to B}$: then $\sigma \in \mathcal{L}_A$ and $\tau \in \mathcal{L}_B$; if $d \in [\![\sigma \to \tau]\!]_\eta$ then:

$$\begin{array}{rll}
e \in [\![\sigma]\!]_\eta & \Rightarrow \rho_A(e) \in [\![\sigma]\!]_\eta & \text{by ind.} \\
& \Rightarrow d(\rho_A(e)) \in [\![\tau]\!]_\eta & \text{by hyp. on } d \\
& \Rightarrow \rho_B(d(\rho_A(e))) \in [\![\tau]\!]_\eta & \text{by ind.}
\end{array}$$

and we conclude since $\rho_{A\to B}(d) = \lambda x.\rho_B(d(\rho_A(x)))$.

Case $\langle \ell_j : \sigma_j \ ^{j\in J}\rangle \in \mathcal{L}_A$, where $A \equiv [\ell_i : B_i \ ^{j\in J}]$ and $J \subseteq I$: then $\sigma_j \in \mathcal{L}_{A\to B_j}$ for all j. This implies that, if $d \in [\![\langle \ell_j : \sigma_j \ ^{j\in J}\rangle]\!]_\eta$ then $d \cdot \ell_j \in [\![\sigma_j]\!]_\eta$ and by induction $\rho_{A\to B_j}(d) \in [\![\sigma_j]\!]_\eta$: the thesis follows since $\rho_A(d) = \langle \ell_i = \rho_{A\to B_i}(d) \ ^{i\in I}\rangle$. ■

Theorem 17 (Soundness of the predicate system w.r.t. retraction models).
If $\Gamma \vdash a^A : \sigma$ then $\Gamma \models a^A : \sigma$.

Proof. By induction on the derivation of $\Gamma \vdash a^A : \sigma$. We show only the interesting cases.

The derivation ends with:

$$(\textit{Type Object})\ \frac{\Gamma, x_i^A : \sigma_i \vdash b_i^{B_i} : \tau_i}{\Gamma \vdash [\ell_i = \varsigma(x_i^A) b_i \ ^{i\in I}]^A : \langle \ell_j : \sigma_j \to \tau_j \ ^{j\in J}\rangle}\ (\forall i \in I \ \wedge \ J \subseteq I)$$

By definition $([\![a^A]\!]_\xi \cdot \ell_j)d = [\![b_j^{B_j}]\!]_{\xi[x_j := \rho_A(d)]}$, where $j \in J \subseteq I$: if $d \in [\![\sigma_j]\!]_\eta$ then, by Lemma 16, $\rho_A(d) \in [\![\sigma_j]\!]_\eta$, since $\sigma_j \in \mathcal{L}_A$; therefore $\xi[x_j := \rho_A(d)] \models \Gamma, x_j^A : \sigma_j$ and, consequently, by induction, $[\![b_j^{B_j}]\!]_{\xi[x_j := \rho_A(d)]} \in [\![\tau_j]\!]_\eta$. This implies that $[\![a^A]\!]_\xi \in [\![\langle \ell_j : \sigma_j \to \tau_j \ ^{j\in J}\rangle]\!]_\eta$.

The derivation ends with:

$$(\textit{Val Select})\ \frac{\Gamma \vdash a^A : \langle \ell_j : \sigma_j \to \tau_j \ ^{j\in J}\rangle \quad \Gamma \vdash a^A : \sigma_k}{\Gamma \vdash (a.\ell_k)^{B_k} : \tau_k}\ (k \in J)$$

The thesis follows immediately by induction: $[\![a^A]\!]_\xi \in [\![\langle \ell_j : \sigma_j \to \tau_j \ ^{j\in J}\rangle]\!]_\eta$ and $[\![a^A]\!]_\xi \in [\![\sigma_k]\!]_\eta$ and by the definition $[\![(a.\ell_k)^{B_k}]\!]_\xi = ([\![a^A]\!]_\xi \cdot \ell_k)[\![a^A]\!]_\xi$.

The derivation ends with:

$$(\textit{Val Update})\ \frac{\Gamma \vdash a^A : \langle \ell_j : \sigma_j \ ^{j\in J}\rangle \quad \Gamma, y^A : \sigma \vdash b^{B_k} : \tau}{\Gamma \vdash (a.\ell_k \Leftarrow \varsigma(y^A) b)^A : \langle \ell_j : \sigma_j \to \tau_k \ ^{j\in J\setminus k}, \ell_k : \sigma \to \tau\rangle}\ (k \in J)$$

Define $c^A \equiv (a^A.\ell_k \Leftarrow \varsigma(y^A) b^{B_k})^A$, and recall that

$$[\![c^A]\!]_\xi = [\![a^A]\!]_\xi \cdot \ell_k := \lambda d.[\![b^{B_k}]\!]_{\xi[y := \rho_A(d)]}.$$

Let $d \in [\![\sigma_j]\!]_\eta$ for some $j \in J$: if $j \neq k$ then $([\![c^A]\!]_\xi \cdot \ell_j)d = ([\![a^A]\!]_\xi \cdot \ell_j)d \in [\![\tau_j]\!]_\eta$ by induction. Otherwise $j = k$ and $([\![c^A]\!]_\xi \cdot \ell_j)d = [\![b^{B_k}]\!]_{\xi[y := \rho_A(d)]} \in [\![\tau_k]\!]_\eta$, again by induction. ■

3.2 The filter model

Definition 18. *A* filter *of predicates is a subset $F \subseteq \mathcal{L}$ of predicates such that:*

1. *$\omega \in F$,*
2. *if $\sigma, \tau \in F$ then $\sigma \wedge \tau \in F$,*
3. *if $\sigma \in F$ and $\sigma \leq \tau$ then $\tau \in F$.*

Let $\mathcal{F}$ be the set of all filters of predicates.

A filter is *principal* if it is of the form $\{\tau \mid \sigma \leq \tau\}$, which we denote by $\uparrow\sigma$ (the upset of σ). As is known from the literature (see e.g. [11]), $\mathcal{F}$ is a λ-model, where

continuous functions, that is mappings $f : \mathcal{F} \to \mathcal{F}$ such that $f(F) = \bigcup_{\sigma \in F} f(\uparrow\sigma)$, are representable by the filters

$$\Psi(f) = \{\sigma{\to}\tau \mid \tau \in f(\uparrow\sigma)\}$$

and functional application is defined by:

$$FG = \{\tau \mid \exists \sigma \in G \,.\, \sigma{\to}\tau \in F\}.$$

Moreover, $\mathcal{F}$ is a solution of the domain equation (1), hence it is a model of the type-free ς-calculus. In the next proposition we spell out the details of the definitions of record selection and record update operations over filters.

Proposition 3. *The following operations on filters interpret the record constant and operations, turning $\mathcal{F}$ into an untyped ς-model:*

1. $\mathsf{emp} = \uparrow\langle\cdot\rangle$;
2. $F \cdot \ell_i = \{\sigma \mid \langle \ell_i : \sigma\rangle \in F\}$;
3. $(F \cdot \ell_i := G) = \{\langle \ell_j : \sigma_j \,^{j \in J}\rangle \mid (j \neq i \ \wedge\ \langle \ell_j : \sigma\rangle \in F) \ \vee\ (j = i \ \wedge\ \sigma_i \in G)\}$.

Proof. The equations of Definition 10 are checked by straightforward calculations. ■

We remark that all the operations above, as well as functional composition, are continuous in their arguments which are filters.

Proposition 4. *$[\![\sigma]\!]_\eta = \{F \in \mathcal{F} \mid \sigma \in F\}$ is a predicate interpretation that satisfies all clauses in Definition 11. Moreover, if $\eta(\kappa) \subseteq [\![K]\!]$ whenever $\kappa \in \mathcal{L}_K$, then $[\![\sigma]\!]_\eta \subseteq [\![A]\!]$ if $\sigma \in \mathcal{L}_A$.* ■

In the following, if X is a variable ranging over filters and $e[X]$ an expression denoting a filter such that the function $\lambda X.e[X]$ is continuous, then we abuse notation writing $\lambda X.e[X]$ for $\Psi(\lambda X.e[X])$.

Lemma 19. *The family $\{\rho_A\}_A$ where $\rho_A(F) = F \cap \mathcal{L}_A$, is a family of retractions turning $\mathcal{F}$ into a retraction model.*

Proof. We check that $F \cap \mathcal{L}_A = \langle \ell_i = \lambda X.(F \cdot \ell_i)(X \cap \mathcal{L}_A) \cap \mathcal{L}_{B_i} \,^{i \in I}\rangle$. Observe that $\sigma \in \langle \ell_i = \lambda X.(F \cdot \ell_i)(X \cap \mathcal{L}_A) \cap \mathcal{L}_{B_i} \,^{i \in I}\rangle$ if and only if $\sigma = \langle \ell_j : \bigwedge \alpha{\to}\beta \,^{j \in J}\rangle$, where $J \subseteq I$ and $\beta \in (F \cdot \ell_i)(\uparrow\alpha \cap \mathcal{L}_A) \cap \mathcal{L}_{B_j}$ for each $\alpha{\to}\beta$ in $\bigwedge \alpha{\to}\beta$ and $j \in J$. On the other hand $\beta \in (F \cdot \ell_i)(\uparrow\alpha \cap \mathcal{L}_A) \cap \mathcal{L}_{B_j}$ if and only if $\langle \ell_j : \alpha'{\to}\beta\rangle \in F \cap \mathcal{L}_A$ for some $\alpha' \in \uparrow\alpha \cap \mathcal{L}_A$.

Now, if $\sigma \in \langle \ell_i = \lambda X.(F \cdot \ell_i)(X \cap \mathcal{L}_A) \cap \mathcal{L}_{B_i} \,^{i \in I}\rangle$ then $\langle \ell_j : \bigwedge \alpha'{\to}\beta \,^{j \in J}\rangle \in F \cap \mathcal{L}_A$ and $\langle \ell_j : \bigwedge \alpha'{\to}\beta \,^{j \in J}\rangle \leq \langle \ell_j : \bigwedge \alpha{\to}\beta \,^{j \in J}\rangle = \sigma$ which is then in $F \cap \mathcal{L}_A$.

Vice versa, if $\sigma \in F \cap \mathcal{L}_A$ then $\sigma = \langle \ell_j : \bigwedge \gamma{\to}\delta \,^{j \in J}\rangle \geq \langle \ell_j : \bigwedge \alpha{\to}\beta \,^{j \in J}\rangle$ for some $J \subseteq I$, $\alpha \in \mathcal{L}_A$, $\beta \in \mathcal{L}_{B_j}$. This implies that $\bigwedge \alpha{\to}\beta \leq \gamma{\to}\delta$ for each $j \in J$ and $\gamma{\to}\delta$ in $\bigwedge \gamma{\to}\delta$. This is true if and only if $\bigwedge Y \leq \delta$ where $Y = \{\beta \mid \alpha \in X\}$ and $X = \{\alpha \mid \alpha \geq \gamma\}$. It follows that $\bigwedge \alpha{\to}\beta \leq \bigwedge X{\to}\bigwedge Y \leq \gamma{\to}\delta$; since $X \subseteq \uparrow\gamma \cap \mathcal{L}_A$ and both filters and languages are closed under finite intersections, $\bigwedge X \in \uparrow\gamma \cap \mathcal{L}_A$, which implies $\delta \in (F \cdot \ell_j)(\uparrow\gamma \cap \mathcal{L}_A)$: now $\sigma \in \langle \ell_i = \lambda X.(F \cdot \ell_i)(X \cap \mathcal{L}_A) \cap \mathcal{L}_{B_i} \,^{i \in I}\rangle$ follows. ■

Theorem 20. *For all a^A such that $E \vdash a^A$, for some E, and all environment ξ such that $\xi \models E$:*

$$[\![a^A]\!]^{\mathcal{F}}_{\xi} = \{\sigma \mid \exists \Gamma.\, \xi \models \Gamma \;\&\; \Gamma \vdash a^A : \sigma\}.$$

Proof. $(\supseteq)$: $(\mathcal{F}, \{\rho_A\}_A)$ is a retraction model by Lemma 19 and therefore, by Theorem 17, if $\xi \models \Gamma$ and $\Gamma \vdash a^A : \sigma$ then $[\![a^A]\!]_\xi \in [\![\sigma]\!]_\eta$, so that $\sigma \in [\![a^A]\!]_\xi$ by definition of $[\![\sigma]\!]_\eta$.

$(\subseteq)$: by induction over a^A.

Case x^A: if $\sigma \in [\![x^A]\!]^{\mathcal{F}}_{\xi} = \xi(x^A) \subseteq \mathcal{L}_A$, then $\{x^A{:}\sigma\}$ is a well formed context, $\xi \models \{x^A : \sigma\}$ and $\{x^A : \sigma\} \vdash x^A : \sigma$ by (*Var*).

Case $a^A \equiv [\ell_i = \varsigma(x^A{}_i) b^{B_i}{}_{i \in I}]^A$: if

$$\sigma \in [\![a^A]\!]^{\mathcal{F}}_{\xi} = \langle \ell_i = \lambda X.[\![b_i^{B_i}]\!]^{\mathcal{F}}_{\xi[x_i := X \cap \mathcal{L}_A]} \cap \mathcal{L}_{B_i}{}_{i \in I} \rangle,$$

then $\sigma = \langle \ell_j : \bigwedge \alpha \to \beta \, {}_{j \in J} \rangle \in \mathcal{L}_A$ for some $J \subseteq I$, where $\beta \in [\![b_j^{B_j}]\!]^{\mathcal{F}}_{\xi[x_j := \uparrow\alpha \cap \mathcal{L}_A]} \cap \mathcal{L}_{B_j}$. By induction hypothesis for each $j \in J$ there exists Γ_j such that $\xi[x_j := \uparrow\alpha \cap \mathcal{L}_A] \models \Gamma_j$ and $\Gamma_j, \vdash b_j^{B_j} : \beta$: this implies that $x^{A_j} : \alpha' \in \Gamma_j$ for some $\alpha' \in \uparrow\alpha \cap \mathcal{L}_A$. Since this holds for all $j \in J$, while clearly $\Gamma_k \vdash b_k^{B_k} : \omega$ for all $k \in I \setminus J$, we derive $\Gamma' \vdash a^A : \langle \ell_j : \bigwedge \alpha' \to \beta \rangle$ by (*Val Object*), where $\Gamma' = \Gamma \setminus x_j^A : \alpha$ for any $j \in J$. Now $\alpha' \geq \alpha$ which implies $\bigwedge \alpha' \to \beta \leq_A \bigwedge \alpha \to \beta$ and we are done.

Case $(a^A.\ell_i)^{B_i}$: if $\tau \in [\![(a^A.\ell_i)^{B_i}]\!]^{\mathcal{F}}_{\xi} = ([\![a^A]\!]^{\mathcal{F}}_{\xi} \cdot \ell_i)[\![a^A]\!]^{\mathcal{F}}_{\xi}$ then there exist $\sigma \in [\![a^A]\!]^{\mathcal{F}}_{\xi}$ such that $\langle \ell_i : \sigma \to \tau \rangle \in [\![a^A]\!]^{\mathcal{F}}_{\xi}$. By induction there are Γ_0, Γ_1 such that $\xi \models \Gamma_i$ for $i = 0, 1$, and $\Gamma_0 \vdash a^A : \langle \ell_i : \sigma \to \tau \rangle$ and $\Gamma_1 \vdash a^A : \sigma$: it follows that $\Gamma = \Gamma_0 \wedge \Gamma_1$ is a well formed context such that $\xi \models \Gamma$, and that $\Gamma \vdash a^A : \langle \ell_i : \sigma \to \tau \rangle$ and $\Gamma \vdash a^A : \sigma$. The thesis follows by (*Val Select*).

Case $(a^A.\ell_i \Leftarrow \varsigma(x^A) b^{B_i})^A$: if

$$\tau \in [\![(a^A.\ell_i \Leftarrow \varsigma(x^A) b^{B_i})^A]\!]^{\mathcal{F}}_{\xi} = [\![a^A]\!]^{\mathcal{F}}_{\xi} \cdot \ell_i := \lambda X.[\![b^{B_i}]\!]^{\mathcal{F}}_{\xi[x := X \cap \mathcal{L}_A]} \cap \mathcal{L}_{B_i},$$

then $\tau = \langle \ell_j : \bigwedge \alpha \to \beta \, {}_{j \in J} \rangle$ for some $J \subseteq I$: if $j \neq i$ then $\langle \ell_j : \bigwedge \alpha \to \beta \rangle \in [\![a^A]\!]^{\mathcal{F}}_{\xi}$, which by induction implies that $\Gamma_j \vdash a^A : \langle \ell_j : \bigwedge \alpha \to \beta \rangle$ for some Γ_j such that $\xi \models \Gamma_j$; if $j = i$ then $\beta \in [\![b^{B_i}]\!]^{\mathcal{F}}_{\xi[x := \uparrow\alpha \cap \mathcal{L}_A]} \cap \mathcal{L}_{B_i}$, hence by induction there exist Γ_i s.t. $\xi \models \Gamma_i, x^A : \alpha$ and $\Gamma_i, x^A : \alpha \vdash b^{B_i} : \beta$. Take $\Gamma = \bigwedge_{j \in J} \Gamma_j$: then $\xi \models \Gamma$ and $\Gamma \vdash a^A : \langle \ell_j : \bigwedge \alpha \to \beta \rangle$ and $\Gamma, x^A : \alpha \vdash b^{B_i} : \beta$, and we conclude by (*Val Update*). ∎

Corollary 1 (Completeness w.r.t. retraction models). $\Gamma \vdash a^A : \sigma \Longleftrightarrow \Gamma \models a^A : \sigma$.

Proof. The 'only if' part is Theorem 17. For the 'if part' define the term environment $\xi_\Gamma(x^B) = \uparrow \tau$ if $x^B : \tau \in \Gamma$, $\uparrow \omega$ if x does not occur in Γ: then $\xi_\Gamma \models \Gamma$, hence $\sigma \in [\![a^A]\!]^{\mathcal{F}}_{\xi_\Gamma}$. By Theorem 20 there exists Γ' such that $\xi_\Gamma \models \Gamma'$ and $\Gamma' \vdash a^A : \sigma$. Now if $\xi_\Gamma \models \Gamma'$ then $\xi_\Gamma(x^B) \in [\![\tau']\!]_\eta$ (for any η consistent with the type interpretation) when $x^B : \tau' \in \Gamma'$; this implies that $\tau' \in \xi_\Gamma(x^B) = \uparrow \tau$, and $x^B : \tau \in \Gamma$: we conclude that $\Gamma \leq \Gamma'$, hence $\Gamma \vdash a^A : \sigma$ by Lemma 8. ∎

4 Conclusions and further work

We have shown that an assignment system of predicates (essentially of intersection types) to typed terms of the object calculus OB_1 induces a sound and complete semantics with respect to a family of models of the ς-calculus using the range of a family of retractions as the interpretation of types. This is a logical semantics, since a retraction model can be constructed in which the denotation of a term coincides with the set (namely the filter) of predicates that can be derived for it in the system.

It remains to be seen how retraction models extend to cope with subtyping and bounded quantification, to model the full ς-calculus. It should be also investigated the relation of retraction models to PER models, which are used in [1] to model the calculus, e.g. along the lines of [13]. This will be the topic of further research.

Acknowledgments

The final version of the paper profited of careful readings and remarks by anonymous referees.

References

1. M. Abadi, L. Cardelli, *A Theory of Objects*, Springer 1996.
2. M. Abadi, G.D. Plotkin, "A Per Model of Polymorphism and Recursive Types", proc. of *IEEE-LICS* 1990, 3355-365.
3. S. Abramsky, "Observation Equivalence and Testing Equivalence", *Theoretical Computer Science* 53, 225–241, 1987.
4. S. Abramsky, "Domain Theory in Logical Form", *APAL* 51, 1991, 1-77.
5. R. Amadio, "Recursion over Realizability Structures", *Info. Comp.* 91, 1991, 55-85. *Theoretical Computer Science*, 102(1):135–163, 1992.
6. S. van Bakel. "Intersection Type Assignment Systems", *Theoretical Computer Science*, 151(2):385–435, 1995.
7. H.P. Barendregt, M. Coppo, M. Dezani, "A Filter Lambda Model and the Completeness of Type Assignment", *JSL* 48, 1983, 931-940.
8. K.B. Bruce, J.C. Mitchell, "PER models of subtyping, recursive types and higher-order polymorphism", proc. of *ACM-POPL* 1992.
9. F. Cardone, "Relational semantics for recursive types and bounded quantification", *LNCS* 372, 1989, 164-178.
10. M. Coppo, M. Dezani, B. Venneri, "Functional characters of solvable terms", *Grund. der Math.*, 27, 1981, 45-58.
11. M. Dezani, E. Giovannetti, U. de' Liguoro, "Intersection types, λ-models and Böhm trees", in [19], 45-97.
12. U. de'Liguoro, "Characterizing convergent terms in object calculi via intersection types", *LNCS* 2044, 2001.
13. U. de'Liguoro, "Subtyping in logical form", in ITRS'02, volume 70.1 of *ENTCS*. Elsevier, 2002.
14. S. Kamin, "Inheritance in Smalltalk-80: a denotational definition", *Proc. of POPL'88*, 1988, 80-87.

15. M. Hennessy, R. Milner, “ Algebraic laws for nondeterminism and concurrency”, *J. of ACM* 32(1), 137–161, 1985.
16. J.L. Krivine, *Lambda-calcul, types et modèles*, Masson 1990.
17. J.C. Mitchell, *Foundations for Programming Languages*, MIT Press, 1996.
18. D. Scott, “Data types as lattices”, *SIAM J. Comput.* 5, n. 3, 1976, 522-587.
19. M. Takahashi, M. Okada, M. Dezani eds., *Theories of Types and Proofs*, Mathematical Society of Japan, vol. 2, 1998.