Subtyping object and recursive types logically

Steffen van Bakel[†] and Ugo de'Liguoro[‡]

[†]Department of Computing, Imperial College, London
[‡]Dipartimento di Informatica, Università di Torino

Subtyping as Subsumption

The basic rule of subtyping is *subsumption*:

$$\frac{E \vdash a{:}A \quad E \vdash A <: B}{E \vdash a{:}B}$$

Subtyping is then a matter of substitution in typed contexts:

$$\frac{E \vdash a : A \quad E \vdash A <: B \quad E, x : B \vdash b[x] : C}{E \vdash b[a] : C}$$

If this is safe then the system is sound.

Semantics of Subtyping

Essentially two approaches:

- inclusion semantics: PER semantics [Buce-Longo, Bruce-Mitchell]
- cohercion semantics [Tannen-Coquand-Gunter-Scedrov]

Problems:

- with PERs: persistent techincal problems while modeling recursive types and bounded quantification;
- with cohercion: too close to syntax, as difficult to use as direct reasoning on term syntax.

We propose a third approach based on *logical semantics* and *domain logic*.

Equational Subtyping

This is the idea of *equational subsumption*:

$$\frac{E \vdash a = a' : A \quad E \vdash A <: B}{E \vdash a = a' : B}$$

If $A<:B$ then the equational theory associated with A is as strong as the theory associated with B, that is:

- any equation holding at A, holds at B as well, the converse does not hold in general;
- dually: if a context $x : B \vdash b[x] : C$ is such that either $b[a] \Downarrow v$ and $b[a'] \Uparrow$ or $b[a] \Downarrow v$, $b[a'] \Downarrow v'$ with $v \neq v' : C$ (a and a' are *separable* at B) then the same holds with $x : A \vdash b[x] : C$.

Two terms are *observationally equivalent* at A if they are not separable at A for any context of ground type.

The case of Records

Consider the following terms representing *records*:

$$r \equiv \{\ell_0 := 0, \ell_1 := 1, \ell_2 := 2\} : \{\ell_0 : \mathit{Int}, \ell_1 : \mathit{Int}, \ell_2 : \mathit{Int}\}$$
$$s \equiv \{\ell_0 := 1, \ell_1 := 1, \ell_2 := 2\} : \{\ell_0 : \mathit{Int}, \ell_1 : \mathit{Int}, \ell_2 : \mathit{Int}\}$$

Then in λ-calculi with records (see e.g. book [Mitchell])

- $r \neq s : \{\ell_0 : \mathit{Int}, \ell_1 : \mathit{Int}, \ell_2 : \mathit{Int}\}$
- $r = s : \{\ell_1 : \mathit{Int}, \ell_2 : \mathit{Int}\}$, because ℓ_0 is not accessible in contexts of type $\{\ell_1 : \mathit{Int}, \ell_2 : \mathit{Int}\}$
- $r = s : \{\ell_1 : \mathit{Int}\}$ as before, which now follows from $\{\ell_1 : \mathit{Int}, \ell_2 : \mathit{Int}\} <: \{\ell_1 : \mathit{Int}\}$

A primer of the (untyped) ς-calculus

Terms (including functional abstraction and application):

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

Reduction rules:

$$\begin{array}{rcll}
[\ell_i = \varsigma(x_i)b_i \ ^{(i \in I)}].\ell_j & \rightarrow & b_j\{x_j \longleftarrow [\ell_i = \varsigma(x_i)b_i \ ^{(i \in I)}]\} & (j \in I) \\
[\ell_i = \varsigma(x_i)b_i \ ^{(i \in I)}].\ell_j \Leftarrow \varsigma(x)b & \rightarrow & [\ell_i = \varsigma(x_i)b_i{}^{i \in I \setminus j}, \ell_j = \varsigma(x)b] & (j \in I) \\
(\lambda x.a)b & \rightarrow & a\{x \longleftarrow b\} & \\
a \ \rightarrow \ b & \Rightarrow & \mathcal{E}[a] \ \rightarrow \ \mathcal{E}[b] &
\end{array}$$

where

$$\mathcal{E}[\_] \quad ::= \quad \_ \mid \mathcal{E}[\_].\ell \mid \mathcal{E}[\_].\ell \Leftarrow \varsigma(x)b \mid \mathcal{E}[\_]a.$$

Examples of reduction

Example 1:

$$\begin{array}{lcl}
[\ell_0 = \varsigma(x_0)1, \ell_1 = \varsigma(x_1)x_1.\ell_0].\ell_1 & \rightarrow & (x_1.\ell_0)\{x_1 \hookleftarrow [\ell_0 = \varsigma(x_0)1, \ell_1 = \varsigma(x_1)x_1.\ell_0]\} \\
& \equiv & [\ell_0 = \varsigma(x_0)1, \ell_1 = \varsigma(x_1)x_1.\ell_0].\ell_0 \\
& \rightarrow & 1\{x_0 \hookleftarrow [\ell_0 = \varsigma(x_0)1, \ell_1 = \varsigma(x_1)x_1.\ell_0]\} \\
& \equiv & 1
\end{array}$$

Example 2:

$$\begin{array}{lcl}
([\ell_0 = \varsigma(x_0)1, \ell_1 = \varsigma(x_1)x_1.\ell_0].\ell_0 \Leftarrow\!= \varsigma(y)2).\ell_1 & \rightarrow & [\ell_0 = \varsigma(y)2, \ell_1 = \varsigma(x_1)x_1.\ell_0].\ell_1 \\
& \rightarrow & [\ell_0 = \varsigma(y)2, \ell_1 = \varsigma(x_1)x_1.\ell_0].\ell_0 \\
& \rightarrow & 2
\end{array}$$

The case of Objects

Consider the ς-terms, where $A \equiv [\ell_0{:}\mathit{Int}, \ell_1{:}\mathit{Int}]$ (from [Abadi-Cardelli 96]):

$$a \equiv [\ell_0 = \varsigma(x_0^A)1, \ell_1 = \varsigma(x_1^A)1]$$
$$b \equiv [\ell_0 = \varsigma(x_0^A)1, \ell_1 = \varsigma(x_1^A)x_1.\ell_0]$$

- $\not\vdash a = b : A$ because they are separated by $x : A \vdash (x.\ell_0 \Leftarrow \varsigma(y^A)0).\ell_1 : \mathit{Int}$
- $\vdash a = b : [\ell_0{:}\mathit{Int}]$ in first order theories and extensions
- $\not\vdash a = b : [\ell_1{:}\mathit{Int}]$, but they can be consistently equated [Gordon-Rees 96]

Logical Semantics

The logical meaning of any a is defined as:

$$[\![a]\!]^{\mathcal{L}} = \text{the set of predicates } \sigma \text{ such that } a \models \sigma.$$

Since equality:

$$a = b : A$$

is modeled by identity of meanings, and it depends on the type A, the logical mening should depend on types:

$$[\![a : A]\!]^{\mathcal{L}} = \text{the set of predicates } \sigma \text{ making sense of } A \text{ s.t. } a \models \sigma.$$

To formalize the idea of "making sense of A" we stratify predicates into a family $\{\mathcal{L}_A\}_A$ of sets, called *languages*.
This is inspired to Abramsky's Domain Logic, where the interpretation of a type A is a domain isomorphic to the filter space generated by the Lindemabaum algebra $(\mathcal{L}_A, \leq_A)$.
The novelty is that there is no accepted domain theoretic interpretation of subtyping, and polymorphism was missing in the Abramsky's work.

Predicates

The set $\mathcal{P}$ of *predicates*, ranged over by $\sigma, \tau, \ldots$ and its subset $\mathcal{P}_S$ of *strict predicates* ranged over by $\phi, \psi, \ldots$, are defined through the grammar:

$$\begin{array}{llll} \mathcal{P}_S & \phi & ::= & \kappa \mid \omega \mid (\sigma \to \phi) \mid \langle \ell{:}\phi \rangle \\ \mathcal{P} & \sigma, \tau & ::= & \phi \mid (\sigma \bigwedge \tau) \end{array}$$

where κ ranges over a countable set of atoms, and $\ell \in \mathbb{L}$ is any (method) label.
The relation $\leq$ over predicates is defined as the last pre-order such that for any $\sigma, \tau \in \mathcal{P}$ and $\phi, \psi \in \mathcal{P}_S$:

1. $\sigma \leq \omega$,
2. $(\sigma \bigwedge \tau)$ is the meet of σ and τ,
3. $(\sigma \to \omega) \leq (\omega \to \omega)$,
4. $\sigma \geq \tau, \phi \leq \psi \Rightarrow (\sigma \to \phi) \leq (\tau \to \psi)$,
5. $\phi \leq \psi \Rightarrow \langle \ell{:}\phi \rangle \leq \langle \ell{:}\psi \rangle$, for any $\ell \in \mathbb{L}$.

Finally $\sigma = \tau \iff \sigma \leq \tau \leq \sigma$, and we write $\sigma < \tau$ if $\sigma \leq \tau$ and $\sigma \neq \tau$.

Predicates versus Intersection Types

Intersection types $\mathcal{IT}$:

$$\sigma, \tau ::= \kappa \mid \omega \mid (\sigma \bigwedge \tau) \mid (\sigma \to \tau) \mid \langle \ell{:}\sigma \rangle.$$

Clearly $\mathcal{P} \subset \mathcal{IT}$, moreover:

Proposition If we extend the definition of $\leq$ to $\mathcal{IT}$ by adding the axiom

$$(\sigma \to \tau) \bigwedge (\sigma \to \tau') \leq \sigma \to (\tau \bigwedge \tau')$$

then for any $\tau \in \mathcal{IT}$ there exists $\sigma \in \mathcal{P}$ such that $\tau = \sigma$.

Remark. Note that, because of the lazyness of the ς-calculus, we have:

$$\omega \to \omega < \omega \text{ and } \langle \ell{:}\omega \rangle < \omega.$$

PT: A Predicate to Type Assignment System

Pre-types and pre-contexts are defined according to the grammar:

$$\begin{array}{rcl} A, B & ::= & X \mid K \mid \mathsf{Top} \mid A{\rightarrow}B \mid \mu X.A \mid \{\ell_i{:}B_i \ \ ^{(i \in I)}\} \mid [\ell_i{:}B_i \ \ ^{(i \in I)}] \\ \Gamma & ::= & \emptyset \mid \Gamma, X{:}\sigma \mid \Gamma, X{:}\sigma{<:}A \end{array}$$

Then we derive sequents of the shape:

$$\Gamma \vdash \diamond \ \text{ and } \ \Gamma \vdash A{:}\sigma$$

meaning "Γ is a well-formed context" and "σ is a predicate making sense of entities of type A under the assumptions in Γ" respectively.
We then define the *language* assosciated to a closed type A as

$$\mathcal{L}_A = \{\sigma \mid \ \emptyset \vdash A{:}\sigma\}.$$

PTt: A Predicate to Typed Terms Assignment System

We introduce a system to derive sequents of the form:

$$\Gamma \vdash a{:}A : \sigma$$

where a is a term of first order ς-calculus, A is a type, σ is a predicate; therefore we need to extend context definition:

$$\Gamma ::= \cdots \mid \Gamma, x{:}A{:}\sigma.$$

Over judgments we can define the projections as follows:

$$(a{:}A{:}\sigma)^{\mathsf{Tt}} = a{:}A, \quad (a{:}A{:}\sigma)^{\mathsf{Pt}} = a{:}\sigma, \quad (a{:}A{:}\sigma)^{\mathsf{PT}} = A{:}\sigma,$$

By extending them to judgments $X{:}\sigma$ and $X{:}\sigma{<:}A$, and componentwise to contexts:

- $\vdash_{\mathsf{Tt}}$, which is the first order ς-calculus,
- $\vdash_{\mathsf{Pt}}$, which is an intersection type system for the ς-calculus,
- $\vdash_{\mathsf{PT}}$, which our tool for defining languages.

Logical Rules

$$\frac{\Gamma \vdash A{:}\sigma \quad \sigma \leq \tau}{\Gamma \vdash A{:}\tau}\,(\leq) \qquad \frac{\Gamma \vdash A{:}\sigma \quad \Gamma \vdash A{:}\tau}{\Gamma \vdash A{:}\sigma \bigwedge \tau}\,(\bigwedge)$$

$$\frac{\Gamma \vdash a{:}A : \sigma \quad \Gamma \vdash a{:}A : \tau}{\Gamma \vdash a{:}A : \sigma \bigwedge \tau}\,(\bigwedge I)$$

Since $\Gamma \vdash A{:}\omega$ if and only if A is a well-formed type expression in the context Γ then for any (closed) type A:

- $\omega \in \mathcal{L}_A$
- $\sigma \in \mathcal{L}_A, \sigma \leq \tau \Rightarrow \tau \in \mathcal{L}_A,$
- $\sigma, \tau \in \mathcal{L}_A \Rightarrow \sigma \bigwedge \tau \in \mathcal{L}_A.$

Arrow Types

$$\frac{\Gamma \vdash A{:}\sigma \quad \Gamma \vdash B{:}\phi}{\Gamma \vdash A{\to}B{:}\sigma{\to}\phi}\,(\to,\to) \qquad \frac{\Gamma \vdash A{:}\sigma \quad \Gamma \vdash B{:}\phi}{\Gamma \vdash A{\to}B{:}\omega}\,(\to,\omega)$$

$$\frac{\Gamma, x{:}A{:}\sigma \vdash a{:}B : \phi}{\Gamma \vdash \lambda x^{A}.a{:}A{\to}B : \sigma{\to}\phi}\,(\to I) \qquad \frac{\Gamma \vdash a{:}A{\to}B : \sigma{\to}\phi \quad \Gamma \vdash b{:}A : \sigma}{\Gamma \vdash ab{:}B : \phi}\,(\to E)$$

The intended meaning of $a{:}A{\to}B{:}\sigma{\to}\phi$ is

> "the term a is a function from entities of type A to entities of type B, such that if the argument satisfies the property σ (pre-condition), then the value satisfies the property ϕ (post-condition)".

For languages we have:

$$\sigma \in \mathcal{L}_A, \phi \in \mathcal{L}_B \Rightarrow \sigma{\to}\phi \in \mathcal{L}_{A\to B}.$$

Recursive Types

$$\frac{\Gamma \vdash \mu X.A{:}\sigma \quad \Gamma, X{:}\sigma \vdash A{:}\phi}{\Gamma \vdash \mu X.A{:}\phi}\,(\mu,\phi) \qquad \frac{\Gamma, X{:}\sigma \vdash A{:}\phi}{\Gamma \vdash \mu X.A{:}\omega}\,(\mu,\omega)$$

The basic intuition is that $\mu X.A \simeq A\{X \longleftarrow \mu X.A\}$ (which is postulated in the subtyping system), and indeed we get:

$$\mathcal{L}_{\mu X.A} = \mathcal{L}_{A\{X \longleftarrow \mu X.A\}}.$$

The following rule is indeed admissible, given the subsumption rule:

$$\frac{\Gamma \vdash a{:}A\{X \longleftarrow \mu X.A\} : \sigma}{\Gamma \vdash a{:}\mu X.A : \sigma}\,(\mu).$$

Predicates in the Language of a Recursive Type

$$\dfrac{\dfrac{X{:}\omega \vdash X \to X{:}\omega \to \omega}{\vdash \mu X.X \to X{:}\omega}\,(\mu,\omega) \qquad \dfrac{X{:}\omega \vdash X{:}\omega \quad X{:}\omega \vdash X{:}\omega}{X{:}\omega \vdash X \to X{:}\omega \to \omega}\,(\to,\to)}{\vdash \mu X.X \to X{:}\omega \to \omega}\,(\mu,\phi)$$

From this it is then not difficult to see that $(\omega \to \omega) \to (\omega \to \omega) \in \mathcal{L}_{\mu X.X \to X}$; but also the unbalanced unfolds (on the predicate side) $(\omega \to \omega) \to \omega$ and $\omega \to (\omega \to \omega)$ are in $\mathcal{L}_{\mu X.X \to X}$, e.g.:

$$\dfrac{\vdash \mu X.X \to X{:}\omega \to \omega \qquad \dfrac{X{:}\omega \to \omega \vdash X{:}\omega \to \omega \qquad \dfrac{X{:}\omega \to \omega \vdash X{:}\omega \to \omega \quad \omega \to \omega \leq \omega}{X{:}\omega \to \omega \vdash X{:}\omega}\,(\leq)}{X{:}\omega \to \omega \vdash X \to X{:}(\omega \to \omega) \to \omega}\,(\to,\to)}{\vdash \mu X.X \to X{:}(\omega \to \omega) \to \omega}\,(\mu,\phi)$$

Record Types

The ς-calculus has no record types. Let us introduce them to explain our subsequent choices. Let $A \equiv \{\ell_i{:}B_i\ ^{(i\in I)}\}$ then:

$$\frac{\Gamma \vdash B_i{:}\phi_i \quad (\forall i \in I)}{\Gamma \vdash A{:}\langle \ell_j{:}\phi_j\rangle}\ (j \in I) \qquad \frac{\Gamma \vdash B_i{:}\phi_i \quad (\forall i \in I)}{\Gamma \vdash A{:}\omega}$$

$$\frac{\Gamma \vdash b_i{:}B_i{:}\phi_i \quad (\forall i \in I)}{\Gamma \vdash \{\ell_i = b_i\ ^{(i\in I)}\}{:}A{:}\langle \ell_j{:}\phi_j\rangle}\ (j \in I) \qquad \frac{\Gamma \vdash b_i{:}B_i{:}\phi_i \quad (\forall i \in I)}{\Gamma \vdash \{\ell_i = b_i\ ^{(i\in I)}\}{:}A{:}\omega}$$

The intended meaning of $a{:}\{\ell_i{:}B_i\ ^{(i\in I)}\}{:}\langle \ell_j{:}\phi\rangle$ is:

> "a is a record having components ℓ_i of type B_i for all $i \in I$; moreover the componet ℓ_j of a enjoys the property ϕ".

Then

$$\phi \in \mathcal{L}_{B_j}, j \in I \Rightarrow \langle \ell_j{:}\phi\rangle \in \mathcal{L}_{\{\ell_i{:}B_i\ ^{(i\in I)}\}}$$

The Language of an Object Type

Let $A \equiv [\ell_i{:}B_i\ ^{(i \in I)}]$:

$$\frac{\Gamma \vdash A{:}\sigma \quad \Gamma \vdash B_i{:}\phi_i \quad (\forall i \in I)}{\Gamma \vdash A{:}\langle \ell_j{:}\sigma \to \phi_j \rangle}\ (j \in I) \qquad \frac{\Gamma \vdash B_i{:}\phi_i \quad (\forall i \in I)}{\Gamma \vdash A{:}\omega}$$

Therefore

$$\sigma \to \phi \in \mathcal{L}_{B_j}, j \in I \Rightarrow \langle \ell_j{:}\sigma \to \phi \rangle \in \mathcal{L}_{[\ell_i{:}B_i\ ^{(i \in I)}]}.$$

Remark. As for record types, a predicate is derived for each component, but just one is selected: this is only to ensure well-formedness of the type expression A; on th other hand predicates might be trivial (i.e. ω), hence it is not restrictive to ask for such subderivations.

Predicates in the Language of an Object Type

Let $A \equiv [\ell_0{:}\mathit{Int}, \ell_1{:}\mathit{Int}]$, and suppose that $\mathsf{O}, \mathsf{E} \in \mathcal{L}_{\mathit{Int}}$ are the predicates of being odd and even integer respectively. Then we can derive $\vdash A{:}\langle \ell_0{:}\omega \to \mathsf{O}\rangle$ as follows (by omitting some obvious inferences):

$$\dfrac{\dfrac{\vdash \mathit{Int}{:}\omega \quad \vdash \mathit{Int}{:}\omega}{\vdash A{:}\omega} \qquad \vdash \mathit{Int}{:}\mathsf{O}}{\vdash A{:}\langle \ell_0{:}\omega \to \mathsf{O}\rangle}$$

Once this is given we can derive more complex statements like:

$$\dfrac{\vdash A{:}\langle \ell_0{:}\omega \to \mathsf{E}\rangle \qquad \dfrac{\vdash A{:}\langle \ell_0{:}\omega \to \mathsf{O}\rangle \quad \vdash \mathit{Int}{:}\mathsf{O}}{\vdash A{:}\langle \ell_1{:}\langle \ell_0{:}\omega \to \mathsf{O}\rangle \to \mathsf{O}\rangle}}{\vdash A{:}\langle \ell_0{:}\omega \to \mathsf{E}\rangle \bigwedge \langle \ell_1{:}\langle \ell_0{:}\omega \to \mathsf{O}\rangle \to \mathsf{O}\rangle}\,(\bigwedge)$$

where the derivation of $\vdash A{:}\langle \ell_0{:}\omega \to \mathsf{E}\rangle$ is similar to that of $\vdash A{:}\langle \ell_0{:}\omega \to \mathsf{O}\rangle$.

Object Types: introduction and elimination

If we had recursive terms, we would stipulate (like [Abramsky]):

$$\frac{\Gamma, x{:}A{:}\sigma \vdash a{:}A : \tau \quad \Gamma \vdash \mathsf{fix} x.a{:}A : \sigma}{\Gamma \vdash \mathsf{fix} x.a{:}A : \tau}$$

Objects are either some kind of recursive records, or records whose field selection is a form of self application [Kamin, Abbadi-Cardelli], hence involving recursion; therefore taking $A \equiv [\ell_i{:}B_i\ ^{(i \in I)}]$:

$$\frac{\Gamma, x_i{:}A{:}\sigma_i \vdash b_i{:}B_i : \phi_i \quad (\forall i \in I)}{\Gamma \vdash [\ell_i = \varsigma(x_i^A) b_i\ ^{(i \in I)}]{:}A : \langle \ell_j{:}\sigma_j \to \phi_j \rangle}\ (j \in I) \qquad \frac{\Gamma \vdash a{:}A : \langle \ell_j{:}\sigma \to \phi \rangle \quad \Gamma \vdash a{:}A : \sigma}{\Gamma \vdash a.\ell_j{:}B_j : \phi}$$

The intended meaning of $a{:}[\ell_i{:}B_i\ ^{(i \in I)}]{:}\langle \ell_j{:}\sigma \to \phi \rangle$ is:

> "a is an object having methods ℓ_i, whose return type is B_i for all $i \in I$; moreover, if the self (namely a itself) satisfies σ (a pre-condition of the method ℓ_j) then the invocation of the method ℓ_j produces a value satisfying ϕ (a post-condition of the method ℓ_j)."

Object Types: Overriding

Bacuse of the previous treatment of object predicates, we can handle method overriding as if it were plain record updating:

$$\frac{\Gamma \vdash a{:}A : \sigma \quad \Gamma, y{:}A{:}\tau \vdash b{:}B_j : \phi}{\Gamma \vdash (a.\ell_j \Leftarrow \varsigma(y^A)b){:}A : \langle \ell_j{:}\tau \to \phi \rangle}\ (\sigma \neq \omega) \qquad \frac{\Gamma \vdash a{:}A : \langle \ell_i{:}\phi \rangle \quad \Gamma, y{:}A{:}\sigma \vdash b{:}B_i : \psi}{\Gamma \vdash (a.\ell_j \Leftarrow \varsigma(y^A)b){:}A : \langle \ell_i{:}\phi \rangle}\ (i \neq j)$$

Note that these are essentially rules for record updating.

Remark. The (decidable) side condition $\sigma \neq \omega$ is needed for derivability of a non trivial predicate (namely $\neq \omega$) to characterize convergence; indeed, while $\langle \ell_i{:}\tau \to \phi \rangle$ is non trivial for any τ and ϕ, it is the case:

$$\frac{a \Downarrow [\ell_i = \varsigma(x_i^{A_i})b_i\ ^{(i \in I)}]}{a.\ell_j \Leftarrow \varsigma(y^A)b \Downarrow [\ell_i = \varsigma(x_i^{A_i})b_i{}^{i \in I \setminus j}, \ell_j = \varsigma(y^A)b]}$$

An example of derivation

Let $\mathsf{O}, \mathsf{E} \in \mathcal{L}_{Int}$ represent the properties of being odd and even integer respectively; then we can derive of $b \equiv [\ell_0 = \varsigma(x_0^A)1, \ell_1 = \varsigma(x_1^A)x.\ell_0]$

$$\dfrac{\dfrac{}{x{:}A{:}\omega \vdash 1{:}\mathit{Int}:\mathsf{O}} \qquad \dfrac{\dfrac{}{x{:}A{:}\langle \ell_0{:}\omega \to \mathsf{E}\rangle \vdash x{:}A:\langle \ell_0{:}\omega \to \mathsf{E}\rangle} \qquad \dfrac{}{x{:}A{:}\langle \ell_0{:}\omega \to \mathsf{E}\rangle \vdash x{:}A:\omega}}{x{:}A{:}\langle \ell_0{:}\omega \to \mathsf{E}\rangle \vdash (x.\ell_0){:}\mathit{Int}:\mathsf{E}}}{\vdash b{:}A:\langle \ell_0{:}\omega \to \mathsf{O}, \ell_1{:}\langle \ell_0{:}\omega \to \mathsf{E}\rangle \to \mathsf{E}\rangle}$$

abbreviating $\langle \ell_0{:}\omega \to \mathsf{O}\rangle \bigwedge \langle \ell_1{:}\langle \ell_0{:}\omega \to \mathsf{E}\rangle \to \mathsf{E}\rangle$ by $\langle \ell_0{:}\omega \to \mathsf{O}, \ell_1{:}\langle \ell_0{:}\omega \to \mathsf{E}\rangle \to \mathsf{E}\rangle$. In the last case, however, the apparently odd predicate we deduce is of use to conclude:

$$\dfrac{\dfrac{\vdots}{\vdash b{:}A:\langle \ell_0{:}\omega \to \mathsf{O}, \ell_1{:}\langle \ell_0{:}\omega \to \mathsf{E}\rangle \to \mathsf{E}\rangle} \qquad \dfrac{}{y{:}A{:}\omega \vdash 2{:}\mathit{Int}:\mathsf{E}}}{\vdash (b.\ell_0 \Leftarrow \varsigma(y^A)2){:}A:\langle \ell_0{:}\omega \to \mathsf{E}, \ell_1{:}\langle \ell_0{:}\omega \to \mathsf{E}\rangle \to \mathsf{E}\rangle}$$

which is what we expected.

The Axiomatization of Subtyping

$$\frac{E \vdash A}{E \vdash A <: A} \qquad \frac{E \vdash A}{E \vdash A <: \mathsf{Top}} \qquad \frac{E \vdash A <: B \quad E \vdash B <: C}{E \vdash A <: C}$$

$$\frac{E', X <: A, E'' \vdash \diamond}{E', X <: A, E'' \vdash X <: A} \qquad \frac{E \vdash A}{E, X <: A \vdash \diamond}\ (X \notin \mathit{dom}(E)) \qquad \frac{E', X <: A, E'' \vdash \diamond}{E', X <: A, E'' \vdash X}$$

$$\frac{E \vdash B_i \quad (\forall i \in I)}{E \vdash [\ell_i{:}B_i{}^{\,i \in I}] <: [\ell_i{:}B_i{}^{\,i \in J}]}\ (J \subseteq I) \qquad \frac{E \vdash A' <: A \quad E \vdash B <: B'}{E \vdash A \to B <: A' \to B'}$$

$$\frac{E \vdash \mu X.A \quad E \vdash A\{X \hookleftarrow \mu X.A\}}{E \vdash \mu X.A <: A\{X \hookleftarrow \mu X.A\}} \qquad \frac{E \vdash \mu X.A \quad E \vdash A\{X \hookleftarrow \mu X.A\}}{E \vdash A\{X \hookleftarrow \mu X.A\} <: \mu X.A}$$

$$\frac{E \vdash \mu X.A \quad E \vdash \mu Y.B \quad E, Y, X <: Y \vdash A <: B}{E \vdash \mu X.A <: \mu Y.B}$$

Subsumption

The subsumption rule now takes the form:

$$\frac{\Gamma \vdash a{:}A : \sigma \quad \Gamma \vdash A{<:}B \quad \Gamma \vdash B{:}\sigma}{\Gamma \vdash a{:}B : \sigma}\,(<:)$$

Theorem Suppose that $\Gamma \vdash A{<:}B$; then:

1. if $\Gamma \vdash B{:}\sigma$ then $\Gamma \vdash A{:}\sigma$,
2. if $\Gamma \vdash a{:}B : \sigma$ and $\Gamma \vdash a{:}A : \omega$ then $\Gamma \vdash a{:}A : \sigma$.

Therefore, if both A and B are closed types, then

$$\vdash A{<:}B \Rightarrow \mathcal{L}_A \supseteq \mathcal{L}_B.$$

The Soundness of Logic Semantics

We define formally the meaning of a term at type A w.r.t. a context E (in the object calculus)

$$[\![a{:}A]\!]_E = \{\sigma \mid \exists \Gamma.(\Gamma)^{\mathsf{Tt}} = E \ \&\ \Gamma \vdash a{:}A : \sigma\}.$$

Nothe that, if $\Gamma \vdash a{:}A : \sigma$ then $\Gamma \vdash A{:}\sigma$; hence, if A is a closed type, this is the same as $\sigma \in \mathcal{L}_A$.

In [Abadi,Cardelli 1996] it is defined an axiomatic theory of object equivalence:

$$E \vdash a \leftrightarrow b : A.$$

Theorem (Soundness) $E \vdash a \leftrightarrow b{:}A \Rightarrow [\![a{:}A]\!]_E = [\![b{:}A]\!]_E$.

Computational Adequacy

Let $a \Downarrow v$ mean "a evaluates to the value v", and $a \Downarrow$ that $a \Downarrow v$ for some v; we say that a and b are *observationally equivalent*, $a \simeq^{\mathcal{O}}_{A} b$ if

$$\forall \text{ground type } K, \text{context } c[\_], \text{value } v.\ \vdash v{:}K\ \&\ \_ : A \vdash c[\_] : K \Rightarrow (c[a] \Downarrow v \Longleftrightarrow c[b] \Downarrow v)$$

Adequacy Theorem

For any closed a of type A, $a \Downarrow$ if and only if there exists $\sigma \in \mathcal{L}_A$ such that $\omega \not\leq \sigma$ (i.e. σ is a non trivial predicate) and $\vdash a{:}A : \sigma$ is derivable.

From this it follows that $[\![a{:}A]\!]_E = [\![b{:}A]\!]_E$ implies $a \simeq^{\mathcal{O}}_{A} b$.

Discussion

Recall that, given $A \equiv [\ell_0{:}\mathit{Int}, \ell_1{:}\mathit{Int}]$,

$$a \equiv [\ell_0 = \varsigma(x_0^A)1, \ell_1 = \varsigma(x_1^A)1]$$
$$b \equiv [\ell_0 = \varsigma(x_0^A)1, \ell_1 = \varsigma(x_1^A)x.\ell_0]$$

which both have type A. Then we have

- $\vdash a{:}A : \langle \ell_1{:}\omega \to 1\rangle$ is derivable while the best we can say about b is $\vdash b{:}A : \langle \ell_1{:}\langle \ell_0{:}\omega \to 1\rangle \to 1\rangle$, hence $[\![a{:}A]\!] \neq [\![b{:}A]\!]$ as it should be;
- we have $[\![a{:}[\ell_0{:}\mathit{Int}]]\!] = [\![b{:}[\ell_0{:}\mathit{Int}]]\!]$: indeed $\vdash a = b : [\ell_0{:}\mathit{Int}]$ is derivable in the ς-calculus equational theory;
- since $\langle \ell_1{:}\omega \to 1\rangle \in \mathcal{L}_{[\ell_1:\mathit{Int}]}$, we also have $[\![a{:}[\ell_1{:}\mathit{Int}]]\!] \neq [\![b{:}[\ell_1{:}\mathit{Int}]]\!]$, which is however sound: we get the desired equality by adding:

$$\frac{\Gamma \vdash a{:}A : \langle \ell : \langle \ell_i{:}\sigma_i \ ^{i\in I}\rangle \wedge \tau \to \rho\rangle \quad \Gamma \vdash a{:}A : \langle \ell_i{:}\sigma_i \ ^{i\in I}\rangle}{\Gamma \vdash a{:}A' : \langle \ell{:}\tau \to \rho\rangle}$$
$$I \cap J = \emptyset,\ A \equiv [\ell_i{:}B_i \ ^{i\in I\cup J}],\ A' \equiv [\ell_i{:}B_i \ ^{i\in J}],\ \langle \ell : \tau \to \rho\rangle \in \mathcal{L}_{A'} :$$

The notion of Model (1)

Let D be a suitable domain, that is such that the untyped ς-calculus can be interpreted in D:

$$[\![a]\!]^D_\rho \in D$$

Then we interpret predicates in the same way as intersection types:

$$[\![\sigma]\!]^D \subseteq D$$

What is the meaning of types? Abstracting from the construction of the assignment system we set:

$$[\![A]\!]^D_\eta \subseteq 2^D$$

namely a type is a family of subsets of D such that

- $D \in [\![A]\!]^D_\eta$
- $S \in [\![A]\!]^D_\eta, S \subseteq T \Rightarrow T \in [\![A]\!]^D_\eta,$
- $S, T \in [\![A]\!]^D_\eta \Rightarrow S \cap T \in [\![A]\!]^D_\eta.$

The notion of Model (2)

Then we can interpret all judgments in the system:

1. $D, \eta \models A{<:}B \Leftrightarrow [\![A]\!]^D_\eta \supseteq [\![B]\!]^D_\eta$,
2. $D, \rho, \eta \models a{:}A \Leftrightarrow [\![a]\!]^D_\rho \in \bigcup [\![A]\!]^D_\eta$,
3. $D, \eta \models A{:}\sigma \Leftrightarrow [\![\sigma]\!]^D \in [\![A]\!]^D_\eta$,
4. $D, \rho, \eta \models a{:}A{:}\sigma \Leftrightarrow [\![a]\!]^D_\rho \in [\![\sigma]\!]^D \in [\![A]\!]^D_\eta$,
5. $D, \rho, \eta \models a \leftrightarrow b : A$ if and only if
   (a) $[\![a]\!]^D_\rho, [\![b]\!]^D_\rho \in \bigcup [\![A]\!]^D_\eta$
   (b) $\forall S \in [\![A]\!]^D_\eta . [\![a]\!]^D_\rho \in S \Leftrightarrow [\![b]\!]^D_\rho \in S$.

By using the filter model technique we can buid a model from the syntax, essentially using predicates and interpreting types as languages. The logical equivalence is the theory of such a model. Moreover it is a PER model: does it coincide with the model in [Abadi, Cardelli 1996]?

The Filter Model

A *filter* is a set $F \subseteq \mathcal{P}$ such that:

1. $\omega \in F$,
2. $\sigma \in F, \sigma \leq \tau \Rightarrow \tau \in F$,
3. $\sigma, \tau \in F \Rightarrow \sigma \bigwedge \tau \in F$.

Let $D = \mathcal{F}$, namely the set of all filters, ordered by inclusion:

1. $[\![a]\!]^{\mathcal{F}}_{\rho} = \{\sigma \mid \exists A, \Gamma.\ \rho \models \Gamma \,\&\, \Gamma \vdash a{:}A : \sigma\}$,
2. $[\![\sigma]\!]^{\mathcal{F}} = \{F \in \mathcal{F} \mid \sigma \in F\}$,
3. $[\![A]\!]^{\mathcal{F}}_{\eta} = \{[\![\sigma]\!]^{\mathcal{F}} \mid \exists \Gamma.\ \eta \models \,\&\, \Gamma \vdash A{:}\sigma\}$.

Theorem. $\mathcal{F}$ is a model w.r.t. the interpretation mapping $[\![]\!]^{\mathcal{F}}$; moreover the logical equivalence is the theory of this model.

The Filter Model vs the PER Model

Is it a PER model? How does $\mathcal{F}$ relates to the model in [Abadi,Cardelli 96]?

Let $\mathcal{A} \subseteq \wp(D)$ be a family of subsets of a given set D; define the binary relation $\mathit{Rel}(\mathcal{A}) \subseteq D \times D$ by setting $(d, e) \in \mathit{Rel}(\mathcal{A})$ if and only if:

1. $d, e \in \bigcup \mathcal{A}$,
2. $\forall S \in \mathcal{A}.\ d \in S \Leftrightarrow e \in S$.

Proposition If $\mathcal{A}$ is a family of subsets of D then $\mathit{Rel}(\mathcal{A})$ is a PER; on the other hand any PER arises form such a family (its quotient).

Open Problem. Is $\mathcal{F}$ isomorphic to a PER model?

Conclusions and further work

We have obtained so far:

- a system which seems to combine nicely the type theory and the logic of a relatively complex calculus, involving functional, recursive and object types, by means of simple tools (the logic is a propositional calculus; type reconstruction and assignment semi-algorithms can be devised in an integrated fashion);
- it gives a syntactic characterization of relevant computational properties (like convergence), and concrete denotational model (a filter model in the sense of [Barendregt-Coppo-Dezani] and Abramsky's Domain Logic, addressing the new theme of polymorphism in that setting.

It seems worthy to work out:

- an extension of the system including second order types and bounded quantification;
- a proper treatment of the store and other imperative features;
- a systematic comparison with Hoare-style logics, like Abadi-Leino logic for object calculi, as well as behavioral subtyping;
- whether authomatic tools can be devised to perform abstract interpretation at type checking time based on the proposed system.

References

Subtyping semantics:

- K. B. Bruce, J. C. Mitchell, “PER models of subtyping, recursive types and higher-order polymorphism”, 1992
- V. Breazu-Tannen, T. Coquand, C. A. Gunter, A. Scedrov, “Inheritance as implicit coercion”, 1991

Object calculi:

- M. Abadi, L. Cardelli, *A Theory of Objects*, 1996

Logical semantics:

- H. P. Barendregt, M. Coppo, M. Dezani, “A Filter Lambda Model and the Completeness of Type Assignment”, 1983
- S. Abramsky, “Domain Theory in Logical Form”, 1991

Previous work by the authors

- U. de’Liguoro, “Subtyping in logical form”, 2002
- S. van Bakel, U. de’Liguoro, “Logical Semantics of the First Order Sigma-Calculus”, 2003
- S. van Bakel, U. de’Liguoro, “Subtyping Objects and Recursive Types Logically”, 2005