Definability and full abstraction in lambda-calculi

Antonio Bucciarelli

Laboratoire Preuves, Programmes et Systèmes
Université Paris Diderot

Outline

Terminology

Language

A typed or untyped λ-calculus endowed with an *operational semantics*, defined via a notion of *reduction* $\rightsquigarrow$, and with a notion of *observational equivalence* $\equiv_{obs}$. The observational equivalence is *contextual* : two terms M and N are equivalent if for any context $C[\,]$, $C[M]$ and $C[N]$ are observably indistinguishable.

Examples :

Language	Reduction	Observation
untyped λ-calculus	β-reduction	head normal forms
PCF	β-δ-Y-reduction	ground constants (integer and booleans)

Hence, in PCF, $M \equiv_{obs} N$ if for all context $C[\,]$ of ground type, $C[M] \rightsquigarrow c$ iff $C[N] \rightsquigarrow c$, c being a ground constant.

In the untyped λ-calculus $M \equiv_{obs} N$ if for all context $C[\,]$, $C[M]$ has a head normal form iff $C[N]$ has a head normal form.

Terminology

Model

A Cartesian closed category, where types of are interpreted by objects, and terms by morphisms. In the untyped case, a model is a reflexive object of the ccc. Convertible terms get the same interpretation : $M \rightsquigarrow N \Rightarrow [\![M]\!] = [\![M]\!]$.

Examples for PCF :

Model	Objects	Morphisms
Scott model	Scott domains	Scott-continuous functions
Stable model	coherence spaces	stable functions

Examples for the untyped λ-calculus :
Graph models, Scott's D_∞.

Semantic brackets $[\![\ ]\!]$ (possibly with superscript : $[\![\ ]\!]^{\textsf{Scott}}$, $[\![\ ]\!]^{\textsf{stab}}$) denote the interpretation of types and terms. For instance, in the Scott's model of PCF :
$[\![\mathtt{bool}]\!] = (\{\bot, \mathit{true}, \mathit{false}\}, \bot < \mathit{true}, \mathit{false})$
$[\![\mathtt{fun}\ (\mathtt{x} : \mathtt{bool}) \to \mathtt{x}]\!] = \{(\bot, \bot), (\mathit{true}, \mathit{true}), (\mathit{false}, \mathit{false})\}$

Full abstraction and definability

L a language, $\mathcal{M}$ one of its models :

Adequacy

- $\mathcal{M}$ is *adequate* for L if, for all terms M, N, $[\![M]\!]^{\mathcal{M}} = [\![N]\!]^{\mathcal{M}} \Rightarrow M \equiv_{obs} N$.
- $\mathcal{M}$ is *fully abstract* for L if, for all terms M, N, $[\![M]\!]^{\mathcal{M}} = [\![N]\!]^{\mathcal{M}} \Leftrightarrow M \equiv_{obs} N$.

Definability

- A morphism f of $\mathcal{M}$ is *L-definable* if there is a closed L-term M such that $[\![M]\!] = f$.
- If all the (finite) elements of $\mathcal{M}$ are L-definable, then (under some reasonable hypothesis) $\mathcal{M}$ is fully abstract for L.

Historical digression

- The λ-calculus, paradigm of the untyped functional languages, was defined by Alonzo Church around 1930. Its first model was found by D. Scott some 40 years later.
- For PCF, paradigm of typed functional languages, the definition of the canonical Scott model, i.e. of the category of Scott domains and Scott-continuous functions, came some years before the precise definition of the language and of its operational semantics (due to Plotkin, around 1975).

Outline

Plotkin's terms

```
let rec omega = fun () -> (omega (): bool);;
(* omega() denotes the undefined boolean value *)

let p = fun (f:bool->bool->bool)->
  if f (omega()) true then
    if f true (omega()) then
      if not(f false false) then true
      else  omega()
    else  omega()
  else  omega();;

let q = fun (f:bool->bool->bool)->
  if f (omega()) true then
    if f true (omega()) then
      if not(f false false) then false
      else  omega()
    else  omega()
  else  omega();;
```

Is there a context allowing to make a difference between p and q ?

The *parallel or* function

$$por\ x\ y = \begin{cases} true & \text{if } x = true \text{ or } y = true \\ false & \text{if } x = false \text{ and } y = false \\ \perp & \text{otherwise} \end{cases}$$

Fact

por is a Scott-continous function.

$[\![p]\!]^{\textbf{Scott}} \neq [\![q]\!]^{\textbf{Scott}}$ since
$[\![p]\!]^{\textbf{Scott}} por = true$ and
$[\![q]\!]^{\textbf{Scott}} por = false$

Theorem (Plotkin)

- The parallel or function is not PCF-definable.
- The terms *p* and *q* (the “parallel or testers”) above are observationally equivalent.
- If PCF is endowed with a new constant computing the parallel or function, then all the finite elements of the Scott model become definable, and the model itself become fully abstract.

Stability

A property shared by all PCF-definable functions, not respected by *por*, is *stability* :
A Scott-continuous function f is stable if for all $x, y : x \uparrow y \Rightarrow f(x \wedge y) = f(x) \wedge f(y)$
where $x \uparrow y$ means $\exists z\ x, y \leq z$.

Stable model (Berry-Girard)

- Objects : coherence spaces.
- Morphisms : stable functions.

In this model, $[\![\mathrm{p}]\!] = [\![\mathrm{q}]\!] = \bot_{(\mathrm{bool} \to \mathrm{bool} \to \mathrm{bool}) \to \mathrm{bool}}$

Nevertheless, the theory of the stable model is not closer to the observational equivalence than the one of the Scott model (they are actually incomparable).

A higher-order example

```
let left_or = fun x y -> if x then true else y;;

let right_or = fun x y -> if y then true else x;;

let or_tester = fun (f: (bool-> bool -> bool) -> bool ) -> bool)
  if f left_or then
    if not(f right_or) then true
    else omega()
  else omega();;
```

In the Scott model, the interpretations of `left_or` and `right_or` are upper bounded by the parallel or. Hence, no functional can yield *true* on the former and *false* on the latter.
As a consequence
$[\![\texttt{or\_tester}]\!]^{\textbf{Scott}} =$
$[\![\texttt{fun}(\texttt{f} : (\texttt{bool} \to \texttt{bool} \to \texttt{bool}) \to \texttt{bool}) \to \texttt{omega}()]\!]^{\textbf{Scott}} = \perp$
On the other hand
$[\![\texttt{or\_tester}]\!]^{\textbf{stab}} F = \mathit{true}$
if $F[\![\texttt{left\_or}]\!]^{\textbf{stab}} = \mathit{true}$ and $F[\![\texttt{right\_or}]\!]^{\textbf{stab}} = \mathit{false}$, and such a functional F does exist in the stable model.
Hence $[\![\texttt{or\_tester}]\!]^{\textbf{stab}} \neq [\![\texttt{fun f} \to \texttt{omega}()]\!]^{\textbf{stab}}$

Toward full abstraction for PCF

Stability is not enough to characterise the definable functions in a purely functional, sequential language like PCF. Further developments :

- Model of sequential algorithms(Berry-Curien).
- Strongly stable model (B.-Ehrhard).
- Game models (Abramsky-Jagadeesan-Malacaria, Hyland-Ong) (first solutions to the full abstraction problem of PCF).

Full abstraction for PCF-like languages :

Language	Model
PCF + por	**Scott**
PCF stable	**stab**
PCF	Games and innocent strategies
PCF + H	Hypercoherences and strongly stable functions
PCF + references (Idealised Algol)	Games and well balanced strategies
PCF + catch (SPCF)	Concrete Data Structures and sequential algorithms
......	......

Outline

The redundant identity

```
let id = fun (x:bool) -> x;;

let r_id = fun x -> if x then x else x;;
```

$[\![id]\!] = [\![r\_id]\!]$ in Scott and stable models.
(hence, *a fortiori*, $id \equiv_{obs} r\_id$).

It is natural to distinguish between these two terms, in order to take into account the usage of resources by a program (intuitively `r_id` uses its argument twice, whereas `id` uses it once.

This boils down to move from *qualitative* models to *quantitative* ones, like the *relational model*.

The category *MRel*

- Objects : sets
- Morphisms : $MRel(A, B) = \mathcal{P}(\mathcal{M}_{fin}(A) \times B)$
  where $\mathcal{M}_{fin}(A)$ denotes the set of finite multi-sets over A, and $\mathcal{P}(A)$ the set of subsets of A.
- Identities : $id_A = \{([\alpha], \alpha) \mid \alpha \in A\}$
- Composition : $f \in MRel(A, B), g \in MRel(B, C)$ $g \circ f =$
  $\{(m_1 \uplus \ldots \uplus m_k, \gamma) \mid \exists \beta_1, \ldots \beta_k \in B, (m_i, \beta_i) \in f, 1 \leq i \leq k, ([\beta_1, \ldots, \beta_k], \gamma) \in g\}$
- Terminal object : $\emptyset$
- Cartesian product : disjoint union
- Function spaces : $B^A = \mathcal{M}_{fin}(A) \times B)$

Fact

MRel is Cartesian closed.

The quantitative flavour of *MRel*

Let $[\![\ ]\!]^{\textbf{rel}}$ denote the interpretation of PCF term in *MRel*. Then :

$[\![\mathtt{id}]\!]^{\textbf{rel}} = \{([\mathit{true}], \mathit{true}), ([\mathit{false}], \mathit{false})\}$

$[\![\mathtt{r\_id}]\!]^{\textbf{rel}} =$
$\{([\mathit{true}, \mathit{true}], \mathit{true}), ([\mathit{false}, \mathit{false}], \mathit{false}), ([\mathit{true}, \mathit{false}], \mathit{true}), ([\mathit{true}, \mathit{false}], \mathit{false})\}$

A reflexive object in *MRel*

The model M_∞

- $M_0 = \emptyset$
- $M_{n+1} = (\mathcal{M}_{fin}(D_n))^{<\omega}$
- $M_\infty = \bigcup_{n\in\omega} D_n$

In particular $M_1 = \{([], [], \ldots, [], \ldots)\}$, call $\star$ the unique element of M_1.
The isomorphism $M_\infty \leftrightarrow M_\infty^{M_\infty}$ is trivial :
$(m_0, m_1, \ldots, m_k, \ldots) \leftrightarrow (m_0, (m_1, \ldots, m_k, \ldots))$.
The interpretation of a closed λ-term in M_∞ coincides with the set of its non-idempotent intersection types.

Full abstraction (without definability)

- M_∞ is fully abstract for the untyped λ-calculus, that is, its theory is the maximal semi-sensible λ-theory $H^\star$.
- Nevertheless, $\star$ is not definable, that is, no closed λ-term is typable with $\star$.

Outline

Toward a resource calculus

Resource calculi are intended to take into account, from an operational point of view, the linear/non linear use of resources (arguments).

Key idea : *linear substitution*
$\mathtt{t}\langle \mathtt{t}'/\mathtt{x}\rangle$ denotes the term $\mathtt{t}$ in which *exactly one* occurrence of $\mathtt{x}$ is replaced by $\mathtt{t}'$.

Example :
$xx\langle \lambda z.z/x\rangle = (\lambda z.z)x + x(\lambda z.z)$

Linear substitution $\Rightarrow$ Non determinism.

The *resource (or differential) λ-calculus* (Ehrhard-Regnier) is an extension of both typed and untyped λ-calculi, featuring linear and classical substitutions.

The untyped resource calculus

Syntax

- Terms
  $t ::= x \mid \lambda x.t \mid t\, b$
- Bags
  $b ::= [t_1, \ldots, t_k, t^!]$

Reduction

$(\lambda x.t)[t_1, \ldots, t_k, t^!] \rightsquigarrow t\langle t_1/x\rangle \ldots \langle t_k/x\rangle\{t/x\}$

Observational equivalence

A term is in outer normal form, if it has no redexes but under a ! ; two terms t, t' are observationally equivalent if for all context $C[\,]$, $C[t]$ reduces to an outer normal form if and only if $C[t']$ reduces to an outer normal form.

As for λ-calculus, the interpretation of terms of the resource calculus in M_∞ may be given via a suitable typing system.

M_∞ and resource calculi

Adequacy

M_∞ is an adequate model of the resource calculus.

Full abstraction

- M_∞ is not fully abstract for the resource calculus (Breuvart, 2013).
- M_∞ is fully abstract for an extension of the resource calculus : the resource calculus with tests. (B.,Carraro,Ehrhard,Manzonetto 2011).

Test elimination

A test elimination procedure allows to give an alternative proof of the full abstraction of M_∞ w.r.t. the untyped λ-calculus, and an original proof of the full abstraction of M_∞ w.r.t. the !-free fragment of the resource calculus.

Outline

Some open problems

- Full abstraction for the resource calculus.
- Full abstraction for the non deterministic λ-calculus.
- Definability and full abstraction for probabilistic PCF.
- Dual problems : given a model, provide an operational characterisation of the theory it induces.
  For instance : provide an operational characterisation of the theory of M_∞ in the resource calculus.

References

- Dana S. Scott : A Type-Theoretical Alternative to ISWIM, CUCH, OWHY. Theor. Comput. Sci. 121(1,2) : 411-440 (1993).
- Gordon D. Plotkin : LCF Considered as a Programming Language. Theor. Comput. Sci. 5(3) : 223-255 (1977).
- Jean-Yves Girard : Linear Logic. Theor. Comput. Sci. 50 : 1-102 (1987).
- Gérard Berry : Stable Models of Typed lambda-Calculi. ICALP 1978 : 72-89.
- Antonio Bucciarelli, Thomas Ehrhard : Sequentiality in an Extensional Framework Inf. Comput. 110(2) : 265-296 (1994)
- Samson Abramsky, Radha Jagadeesan, Pasquale Malacaria : Full Abstraction for PCF. Inf. Comput. 163(2) : 409-470 (2000)
- J. M. E. Hyland, C.-H. Luke Ong : On Full Abstraction for PCF : I, II, and III. Inf. Comput. 163(2) : 285-408 (2000).
- Thomas Ehrhard, Laurent Regnier : The differential lambda-calculus. Theor. Comput. Sci. 309(1-3) : 1-41 (2003)
- Antonio Bucciarelli, Thomas Ehrhard, Giulio Manzonetto : Not Enough Points Is Enough. CSL 2007 : 298-312

- Antonio Bucciarelli, Thomas Ehrhard, Giulio Manzonetto : Categorical Models for Simply Typed Resource Calculi. Electr. Notes Theor. Comput. Sci. 265 : 213-230 (2010).
- Antonio Bucciarelli, Alberto Carraro, Thomas Ehrhard, Giulio Manzonetto : Full Abstraction for Resource Calculus with Tests. CSL 2011 : 97-111
- Flavien Breuvart : The resource lambda calculus is short-sighted in its rezlational model. To appear (TLCA 2013).