Denotational semantics, normalisation, and the simply-typed $\lambda\mu$-calculus

Vladimir Komendantsky

INRIA – Sophia Antipolis

22 June 2007

Contents

Contents

Denotational semantics for a lambda calculus

- Lambda calculus is commonly known as a theory of computable functions.
- The relationship of this theory to actual functions, e.g. functions between sets, is established by means of a suitable denotational semantics.
- Denotational semantics gives meaning to a language by assigning mathematical objects as values to its terms.
- The $\lambda\mu$-calculus is a lambda calculus with additional constructs for representing the constructive content of classical proofs and handling continuations.
- We introduce a (strong) normalisation method for simply-typed $\lambda\mu$-terms that is obtained by constructing an inverse of the semantic evaluation functional. The method is inspired by that of Berger & Schwichtenberg (1991).

Typing rules of the $\lambda\mu$-calculus

$$\frac{}{\Gamma \vdash x : A \mid \Delta} \quad \text{if } x{:}A \in \Gamma \qquad\qquad \frac{}{\Gamma \vdash c^A : A \mid \Delta}$$

$$\frac{\Gamma \vdash M : B^A \mid \Delta \qquad \Gamma \vdash N : A \mid \Delta}{\Gamma \vdash MN : B \mid \Delta} \qquad\qquad \frac{\Gamma, x{:}A \vdash M : B \mid \Delta}{\Gamma \vdash \lambda x^A.M : B^A \mid \Delta}$$

$$\frac{\Gamma \vdash M : A \mid \Delta}{\Gamma \vdash [\alpha]M : \bot \mid \Delta} \quad \text{if } \alpha{:}A \in \Delta \qquad\qquad \frac{\Gamma \vdash M : \bot \mid \alpha{:}A, \Delta}{\Gamma \vdash \mu\alpha^A.M : A \mid \Delta}$$

$$\frac{\Gamma \vdash M : A \mid \Delta}{\Gamma' \vdash M : A \mid \Delta'} \quad \text{if } \Gamma \subseteq \Gamma',\ \Delta \subseteq \Delta'$$

Decision problem for the $\lambda\mu$-calculus

For any possibly open $\lambda\mu$-terms M and N of type A, decide whether $\Gamma \vdash M = N : A \mid \Delta$, where $=$ denotes the equality of $\lambda\mu$-terms in context.

With each $\lambda\mu$-term M we associate its *abstract normal form* $\mathrm{nf}(M)$, for which there exists a reverse function fn from normal forms to terms such that

(NF1, completeness) $\Gamma \vdash \mathrm{fn}(\mathrm{nf}(M)) = M : A \mid \Delta$

(NF2, soundness) $\Gamma \vdash M = N : A \mid \Delta$ implies $\mathrm{nf}_{\Gamma,\Delta}(M) \equiv \mathrm{nf}_{\Gamma,\Delta}(N)$

Note

nf is allowed not to be injective and hence there is no *inverse* function nf^{-1} in general.

Why this gives a semantics of *normalisation*

The conditions (NF1) and (NF2) imply the soundness and completeness property:

$$\Gamma \vdash M = N : A \mid \Delta \quad \text{iff} \quad \mathrm{nf}_{\Gamma,\Delta}(M) \equiv \mathrm{nf}_{\Gamma,\Delta}(N)$$

Connections to the continuation semantics

(Strachey & Wadsworth 1974)
Continuation semantics is a method to formalise the notion of a control flow in programming languages. Any term is evaluated in a context which represents the "rest of computation". Such context is called *continuation*.

(Lafont, Streicher & Reus 1993; Hofmann & Streicher 1998; Selinger 1999)
By the call-by-name continuation passing style translation, a judgement of the $\lambda\mu$-calculus

$$x_1{:}B_1, \ldots, x_n{:}B_n \vdash M : A \mid \alpha_1{:}A_1, \ldots, \alpha_m{:}A_m \tag{1}$$

is translated to the judgement of the $\lambda^{R\times}$-calculus

$$x_1{:}C_{B_1}, \ldots, x_n{:}C_{B_n}, \alpha_1{:}K_{A_1}, \ldots, \alpha_m{:}K_{A_m} \vdash \underline{M} : C_A \tag{2}$$

Call-by-name continuation passing style translation

The taget calculus of the CPS translation has a pair of types

K_A – the type of continuations of type A
C_A – the type of computations of type A

for each type A of the $\lambda\mu$-calculus, defined as follows:

$$\begin{array}{rcll} K_\sigma & = & \sigma & \text{where } \sigma \text{ is a type constant} \\ K_{B^A} & = & C_A \times K_B & \\ K_\perp & = & 1 & \\ C_A & = & R^{K_A} & \end{array}$$

The CPS translation is defined by means of inductive rules.

Contents

Complete partial orders

A subset P of a partial order is *directed* if every finite subset of P has an upper bound in P.

A *complete partial order* (*cpo*) is a partial order having least upper bounds (lubs) of all directed subsets but not necessarily a least element.

A function between two cpos is *Scott-continuous* if it preserves lubs of directed sets.

A *pointed cpo* (also, *domain*) is a cpo that has also a least element called the *bottom element*.

By B^A we mean the space of Scott-continuous functions from A to B.

Negated domains

A *negated domain* is an object of the form R^A, where A is a cpo and R is some chosen domain (= pointed cpo) of "responses". The domain $R \cong R^1$ is the meaning of the proposition $\bot$ (false). The denotation of a λ-term is an object of a domain R^A mapping elements of A (continuations) to elements of R (responses/answers).

Remark

In order to guarantee for negated domains parametrised by R to have a least-fixed-point operator, one should assume that R has a least element.

Continuation semantics in the setting of negated domains

Due to isomorphism $(R^B)^{R^A} \cong R^{R^A \times B}$, the cpo of continuations for the exponential $(R^B)^{R^A}$ is $R^A \times B$, which means that a continuation for a function f from R^A to R^B is a pair $\langle d, k \rangle$, where $d \in R^A$ is an argument for f and $k \in B$ is a continuation for $f(d)$.
Negation is defined as $\neg R^A := R^A \Rightarrow R^1$. We have
$\neg R^A \cong R^{R^A \times 1} \cong R^{R^A}$.
There is a canonical map from $R^{R^{R^A}}$ to R^A which provides an interpretation of the classical law $\neg\neg P \Rightarrow P$ (reductio ad absurdum). This interpretation can be assigned as meaning to the control operator $\mathcal{C}$ of $\lambda\mathcal{C}$-calculus (Felleisen 1986; Griffin 1990).

Interpretation of $\lambda\mu$-calculus lexical constructs

- Naming $[\alpha]M$ is interpreted as application of the meaning of M (that is an element of a domain R^A) to the continuation bound to α (that is an element of a cpo A) thus resulting in an element of R.
- μ-abstraction $\mu\alpha.M$ is interpreted as functional abstraction over the continuation variable α at the level of continuation semantics.

Contents

Long $\beta\eta$-normal form

The set of λ-terms in long $\beta\eta$-normal form is inductively defined by

$$(xM_1 \dots M_n) : \sigma \qquad\qquad \lambda x.M$$

where σ is a base type.

The idea of our method is to compute the long $\beta\eta$-n.f. by evaluating a $\lambda\mu$-term in an appropriate continuation model.

Sets equipped with partial equivalence relations

A *per-set* A is a pair $A = (|A|, \sim_A)$, where $|A|$ is a set and $\sim_A$ is a partial equivalence relation (per), that is a symmetric and transitive relation, on $|A|$.
A *per-function* between per-sets $A = (|A|, \sim_A)$ and $B = (|B|, \sim_B)$ is a function $f : |A| \to |B|$ such that $a \sim_A a'$ implies $f(a) \sim_B f(a')$, for all $a, a' \in |A|$.

Obtaining the semantics of normalisation

1. Consider a simple denotational semantics for $\lambda\mu$-calculus with a given signature (base types and constants) and a fixed response object R.
2. Annotate interpretations of contexts and terms by sequences of object/control variables.
3. Relate interpretations of $\beta\eta$-convertible terms by a per.
4. Construct an annotated canonical model (and hence find the canonical interpretation of the $\lambda\mu$-calculus in that model).
5. Consider two naturally isomorphic interpretations: the presheaf interpretation of the canonical model by the Yoneda embedding, and the interpretation freely extending the interpretation of objects of the canonical model by the Yoneda embedding.
6. The normalisation function can then be obtained by “dipping” the free presheaf interpretation of a $\lambda\mu$-term into the natural isomorphism above.

Soundness and completeness of the normalisation function

Theorem (Completeness, NF1)

There is a function fn *from abstract normal forms to terms such that, for a well-typed* $\lambda\mu$*-judgement* $\Gamma \vdash M : C \mid \Delta$*,*

$$\Gamma \vdash \mathrm{fn}(\mathrm{nf}(\llbracket \Gamma \vdash M : C \mid \Delta \rrbracket^0)) = M : C \mid \Delta$$

is a valid equation of the $\lambda\mu$*-calculus.*

Theorem (Soundness, NF2)

For a valid equation $\Gamma \vdash M = N : C \mid \Delta$ *of the* $\lambda\mu$*-calculus, it holds that*

$$\mathrm{nf}(\llbracket \Gamma \vdash M : C \mid \Delta \rrbracket^0) \equiv \mathrm{nf}(\llbracket \Gamma \vdash N : C \mid \Delta \rrbracket^0) \;.$$

Contents

Inductive rules for the call-by-name continuation passing style translation

$$
\begin{array}{rcl}
\underline{x} & = & \lambda k^{K_A}.xk \quad \text{where } x : A \\
\underline{c^A} & = & \lambda k^{K_A}.ck \\
\underline{MN} & = & \lambda k^{K_B}.\underline{M}\langle \underline{N}, k\rangle \quad \text{where } M : B^A,\ N : A \\
\underline{\lambda x^A.M} & = & \lambda \langle x, k\rangle^{K_{B^A}}.\underline{M}k \quad \text{where } M : B \\
\underline{[\alpha]M} & = & \lambda k^{K_\perp}.\underline{M}\alpha \quad \text{where } M : A \\
\underline{\mu\alpha^A.M} & = & \lambda \alpha^{K_A}.\underline{M} * \quad \text{where } M : \perp
\end{array}
$$