Strong Call-by-Need

Thibaut Balabonski    Delia Kesner
(LRI, Université Paris-Sud)  (IRIF, Université Paris-Diderot)

Functional programming languages may use different evaluation orders, also called reduction strategies: for instance, ML and OCaml are driven by call-by-value strategies in which the parameters of a function call are evaluated prior to the call itself, while Haskell obeys to a call-by-need strategy (also called lazy strategy) in which the parameters are evaluated only when they are needed (which possibly means never). The standard model of computation in such languages is restricted to weak reduction, which does not compute inside abstractions (a.k.a. functions): the aim of evaluation in a functional programming language is to produce a value, and since abstractions are already considered to be values there is no point in pursuing computation inside them.

In constrast to weak reduction, strong reduction [4] evaluates inside the bodies of abstractions in order to reach the so-called normal forms. This evaluation model is more complex since it entails in particular the evaluation of open terms containing free variables, but it is required in applications relying on symbolic computation, as for example program specialization by partial evaluation, or it is used in the implementation of proof assistants based on type theory (for instance Agda or Coq).

In [2] a strong call-by-need strategy is defined for the lambda-calculus. This strategy, which defines a deterministic function on lambda-terms, computes strong normal forms while respecting the spirit of call-by-need strategies: arguments are evaluated only if they are needed, and at most once to avoid duplications of work. The strong call-by-need strategy is conservative w.r.t. the so-called weak call-by-need strategy [1], defined to formally model lazy functional programming languages. Moreover, the strong call-by-need strategy is proved to be normalising, i.e. it finds the normal form of any weakly normalising term (a term that admits a normal form even though its evaluation may diverge along some strategies).

The goal of this internship is to explore an adequate notion of non-deterministic strong call-by-need calculus, being as general as possible but capturing the (deterministic) call-by-need strategy defined in [2] and retaining the essential properties of call-by-need.

The calculus is expected to have several properties, which need to be proved: it should be confluent, it should reduce only needed redexes, and it should respect a refined notion of preservation of weak-normalisation (as long as a term is weakly normalising in the lambda-calculus, we wish that each possible strategy in the strong call-by-need calculus reaches that normal form). As a result, this calculus is expected to be the proper setting in which alternative and optimized strong call-by-need strategies can be expressed and studied.
Mathematical tools such as rewriting theory and (intersection) type systems (see for example [3]) will be used for this internship.

Contact

• Delia Kesner, Professor, Université Paris-Diderot.
  E-mail: kesner@irif.fr
  URL: http://www.irif.fr/~kesner

• Thibaut Balabonski, Maître de Conférences, Université Paris-Sud.
  E-mail: blsk@lri.fr
  URL: https://www.lri.fr/~blsk/

Duration

Six months.

Place

IRIF, Bâtiment Sophie Germain, 5 rue Thomas Mann. 75205 Paris CEDEX 13.

References


