Operational Game Semantics for OCaml Type System

Guilhem Jaber
Gallinette - LS2N - Université de Nantes
guilhem.jaber@inria.fr - http://guilhem.jaber.fr

Keywords: Functional Programming, Type System, Semantics of Programming Languages

1 Context

Reasoning on higher-order programs with imperative features like mutable memory or exceptions, is a challenging problem. A key aspect is *compositionality*, that allows modular reasoning on source code, analysing each function or each module independently, then combining the results. It is crucial in order to design techniques that scale to large codebases.

OCaml type system enforces strong properties on programs, which guarantees the absence of a large class of bugs, following Milner's slogan "well-typed programs cannot go wrong".

The goal of this internship is to design compositional semantic reasoning over OCaml programs that takes into account the safety properties enforced by its type system. We will focus on two aspects of the OCaml type system, Hindley-Milner polymorphism and (generalized) algebraic data types.

2 Subject

We will develop *operational game semantics*, that provides a general framework to study such effectful typed higher-order languages [Lai07]. To do so, it represents programs as Labelled Transition Systems (*LTS*) that generate traces that correspond to interaction with any possible environment. Doing so, it provides a compositional representation of programs, by specifying the possible behaviors of the interacting environments.

In [JT16], an operational game model was developed for a language with Church-style polymorphism and references. Being Church-style, no "value restriction", a condition needed to enforce soundness of the type system in presence of imperative features [Wri95], was necessary.

The first goal of this internship is to adapt this semantics to a Curry-style language, with Hindley-Milner `let`-polymorphism, and value restriction. We will also consider records, including the possibility of encoding higher-rank polymorphism with them.

The next step will be to extend this model to *generalized algebraic data types* (GADT). They provide a way to enforce strong invariants on the data-types handled by programs. We will use the work of [SCJD07] to start with a specification of GADT represented as System F_C, corresponding to the extension of System F with type equality coercions. The notion of type disclosure introduce in [JT16] could be a first step to represent these type equality coercions in operational game semantics

3 Expected Skills

We are looking for candidates with good skills in functional programming (ideally OCaml), and knowledge of semantics of programming languages and type systems.

4 Practical Informations

This internship is part of the "Compositional Automated Vericiation of OCaml" (CAVOC) project funded by the Inria/Nomadic Labs partnership. The intern will receive a stipend ("gratification de stage") following the legal rate.

It may lead to a PhD thesis on the general topic of automated verification of OCaml programs.

It will be located in the Inria Gallinette team of LS2N, on the Faculté des Sciences campus of the university of Nantes.

References

[JT16] Guilhem Jaber and Nikos Tzevelekos. Trace semantics for polymorphic references. In *Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science*, LICS '16, page 585–594, New York, NY, USA, 2016. Association for Computing Machinery.

[Lai07] Jim Laird. A fully abstract trace semantics for general. In *Proceedings of the 34th International Conference on Automata, Languages and Programming*, ICALP'07, page 667–679, Berlin, Heidelberg, 2007. Springer-Verlag.

[SCJD07] Martin Sulzmann, Manuel MT Chakravarty, Simon Peyton Jones, and Kevin Donnelly. System F with type equality coercions. In *Proceedings of the 2007 ACM SIGPLAN international workshop on Types in languages design and implementation*, pages 53–66, 2007.

[Wri95] Andrew K Wright. Simple imperative polymorphism. *Lisp and symbolic computation*, 8(4):343–355, 1995.