Scientific Report

Sergueï Lenglet

20/09/2010-19/09/2011

Host: PLERCIM - Wrocław
Scientific coordinator: Leszek Pacholski

1 Scientific Activity

Static analysis allows to formally prove properties of programs without having to execute them. Various properties can be proved using different techniques, e.g., type systems guarantee check well-formedness of programs or their termination, and bisimulations can be use to compare the behaviors of programs. During my fellowship, I studied such techniques for the λ-calculus extended with control operators. Such operators allow programs to manipulate their execution contexts, and can be used, e.g., to implement exceptions, backtracking algorithms, etc.

Type systems. The first part of my fellowship was dedicated to the definition of an expansion mechanism for universal quantifiers. Expansion is an operation on typings (i.e., pairs of typing environments and result types) defined originally in type systems for the λ-calculus with intersection types in order to obtain principal (i.e., most informative, strongest) typings. In a type inference scenario, expansion allows postponing choices for whether and how to use non-syntax-driven typing rules (e.g., intersection introduction) until enough information has been gathered to make the right decision. With J. B. Wells, we have extended expansion to a system with $\forall$-quantifiers (System F). Our system turns type inference into a constraint solving problem; this could be helpful to design a modular type inference algorithm for System F types in the future. This work has been submitted for publication at a conference [1].

I then worked on a comprehensive study of a type system for control operators defined in the continuation-passing style (CPS) hierarchy. CPS is a particular programming style where all recursive calls are tail calls. Programming in this style requires to pass around an additional parameter, the continuation, which represents the rest of the computation. To use the power of CPS in direct style (i.e., without having to carry an extra parameter), one can use control operators, defined using CPS. When CPS is iterated, it generates what is called the CPS hierarchy, which can be used to defined more expressive control operators. With Małgorzata Biernacka and Dariusz Biernacki, we have defined a type system for control operators defined in the CPS hierarchy, and proved various properties, such as subject reduction, and termination of evaluation. This work has been accepted at the conference PPDP 2011 [3].

Behavioral equivalences. Finally, the last part of my fellowship was dedicated to the study of behavioral equivalences for a λ-calculus with delimited control operators. Two terms are said contextually equivalent when we can replace one for the other in any program without changing the behavior of the whole program. In general, the behavior of a program is specified using a set of observable actions; in the λ-calculus, we only observe termination. This notion is widely accepted as the most natural behavioral equivalence on terms, but is hard to use in practice, because we have to test terms within all possible programs. Therefore it is common to look for easier-to-use equivalences, called bisimilarities. Up to now, there were very few works on the behavioral theory of languages with control operators. With Dariusz Biernacki, we have defined an easy-to-use bisimilarity for a λ-calculus with delimited-control operators, and proved that is relates exactly the same terms as contextual equivalence. We have also studied the relationship between our bisimilarity and another equivalence on terms based on CPS. This work has been submitted to a conference [2].

References

[1] Sergueï Lenglet J. B. Wells. Expansion for universal quantifiers. Submitted.

[2] Dariusz Biernacki Sergueï Lenglet. Bisimilarities for delimited-control operators. Submitted.

[3] Malgorzata Biernacka, Dariusz Biernacki, and Sergueï Lenglet. Typing control operators in the cps hierarchy. In *Proceedings of the 13th international ACM SIGPLAN symposium on Principles and practices of declarative programming*, PPDP '11, pages 149–160, New York, NY, USA, 2011. ACM.

2 Attended Seminars, Workshops, and Conferences

During my fellowship, I attended the following events:

- Types Workshop, Warsaw, 13-16 October 2010
- Theory and Practice of Delimited Continuations (workshop), Novi Sad, 29-30 May 2011
- Typed Lambda-Calculus and Application, Novi Sad, 1-3 June 2011
- Principles and Practice of Declarative Programming, Odense, 20-22 July 2011

3 Research Exchange Programme

GALLIUM team, INRIA Rocquencourt, 3-7 January 2011. Host: Didier Rémy. The GALLIUM team has developed an expertise in types for functional languages such as Ocaml. In particular, Didier Rémy is working on a conservative extension of ML, called ML^F, with the same expressive power as System F. Assuming some type annotations are provided, the type inference for ML^F is decidable. ML^f is closely related to my work on the expansion for System F. Visiting Didier Rémy gave me the opportunity to compare our type systems, and to have a better understanding of their respective strengths and weaknesses.

SARDES Team, INRIA Rhônes-Alpes, 4-8 July 2011. Host: Alan Schmitt. A part of the SARDES team works on the development of static analysis methods for concurrent systems, and in particular on the development of behavioral equivalences for process calculi. The bisimilarities and proof techniques that I have used for delimited-control operators are similar to the ones used for these calculi. The visit of the SARDES team was the occasion to discuss the relationship between process calculi and λ-calculi with delimited-control operators. Because delimited-control operators allow for non-deterministic behaviors, it might be possible to encode a process calculus, e.g., the π-calculus, into a λ-calculus with control operators.