ALMA MATER STUDIORUM — UNIVERSITA' DI BOLOGNA
DIPARTIMENTO DI INFORMATICA – SCIENZA E INGEGNERIA
MURA ANTEO ZAMBONI N. 7 40126 BOLOGNA
Tel. +39 051 20 94873 – Fax +39 051 20 94510

Report on the thesis "Reasoning About Control Effects: From Program Equivalence to Language Design", by Piotr Polesiuk

May 25, 2020

The general topic of the thesis is formal method for reasoning about programming languages. Specifically the thesis focuses on programming languages equipped with control operators that allow a programmer to manipulate computational effects, including for instance input/output, mutable references, exceptions, backtracking. Some of these effects are very powerful, requiring capture and management of the context surrounding the current computational point in a program. Control operators can be divided into two categories, both considered in the thesis: delimited control operators and handlers of algebraic effects. Control operators have remarkable expressive power, and are finding a growing space in modern programming languages. Only fairly recently they are undergoing a careful scrutiny within the formal method community. Formal methods for languages with control operators are challenging because of their expressiveness and because of the intricacies in the way they manipulate the continuation points. All these aspects are strong justifications and motivations for the work in the thesis.
The techniques proposed in the thesis range from coinduction and bisimulation to type systems and logical relations. The insights gained from the study of the techniques are injected into the design of new programming language constructs with control operators, notably the Helium languages, designed and implemented during the thesis work. Extensive use throughout the thesis is made of the Coq proof assistant for mechanically checking proofs.
The thesis consists of about 50 pages of overview, where control operators are explained, using several examples, and the contents and results in the thesis are outlined. Then there are 9 chapters, each of them representing a published paper. The first 4 chapters have a clear focus towards bisimulation and untyped languages. The last 5 chapter have a focus towards logical relations and type systems. The thesis presents very many interesting contributions. It would be long to recall all of them here, thus below I only mention a few of them.

- Refinement (in fact a few refinements) of environmental bisimulation and Madiot's framework to handle delimited control operators.

- Developments of the theory of normal form (or open) bisimulation, a form of bisimulation for higher-order calculi that allows one to directly reason on terms with free variables; such developments notably concern the justification of eta-expansion (i.e., extensionality) and the corresponding 'up-to context' enhancement; a few calculi are considered, with and without delimited control operators, to witness the robustness of the methods.

- A full abstraction result for bisimulation in a lambda calculus with local references; the result is beautiful and ingenious, requiring careful twists to the standard definition of bisimulation.

- An abstract formulation of the new bisimulation clauses introduced in the previous work, on complete lattices; this allows one both to better understand the clauses and to apply them more easily to other languages.

- Logical relations that, technically, are heterogeneous, biorthogonal, and step-indexed, to prove the coherence of coercion semantics of languages with subtyping.

- A refinement of the logical relation technique to a setting with row polymorphic algebraic effects; besides providing a reasoning method for establishing equivalences between program terms, the study has also led to identifying a new elegant programming construct to manage multiple occurrences of the same effect in a row.

- Proposals for abstraction in languages with algebraic effects, of the kind provided by modules in mainstream programming languages; two such mechanisms are studied, one for existential effects, to hide the details of a given effect, the other for local effects, to avoid interference on a given effect from code coming from the outside; these features are studied in a dedicated calculus, the core of the new experimental programming language Helium.

- A solution to the problem of combining different uses of the same effect in a program; in the solution, an instance of an algebraic effect is a lexically scoped variable; two styles of operational semantics for the resulting language are studied and compared, also introducing for this a new Kripke-style logical relation technique; a major issue here was understanding parametric polymorphism in presence of effects.

- Identification of the form of polymorphism needed to relate effect handlers and delimited control operators.

I found the thesis truly remarkable, both in quantity (number of results, number of publications) and in quality (originality, depth and breath of the contributions). The exceptional quality is also witnessed by the venues of the publications, including 3 LMCS, 3 POPL, 1 FOSSACS — very top venues for papers in programming languages. I was highly impressed by the technicalities of the contributions, often very non-trivial, and yet formalized and carried out in a very elegant manner. I was also very impressed by the interplay between theory and practice: each chapter presents technical results, but their practical impact is always carefully evaluated. This shows up both in proofs of concrete examples for existing programming languages, and (less obviously) in proposals of new programming constructs, or even

entire new calculi or languages. The title itself of the thesis, as well as the 'motto' at the beginning of the introduction, well illustrate the interplay between theory and practice that is found in the document.
In summary, I consider this thesis clearly above the average level of a PhD thesis. I would like to congratulate both the author and his PhD advisor, as well as the collaborators, for the work done. I also strongly recommend the Scientific Council to vote the thesis as outstanding (or excellent, I am not sure about the terminology).

[illegible]

Prof. Davide Sangiorgi
Departmento di Informatica (DISI)
Universita' di Bologna
`sangio@cs.unibo.it`
`http://www.cs.unibo.it/~sangio/`