Intersection Types for Normalization and Verification

Kazushige Terui

RIMS, Kyoto University
Kitashirakawa Oiwakecho, Sakyo-ku, Kyoto 606-8502, Japan
terui@kurims.kyoto-u.ac.jp

Abstract

One of the basic principles in typed lambda calculi is that typable lambda terms are normalizable. Since the converse direction does not hold for simply typed lambda calculus, people have been studying its extensions. This gave birth to the intersection type systems, that exactly characterize various classes of lambda terms, such as strongly/weakly normalizable terms and solvable ones (see e.g. [6] for a survey).

More recently, a new trend has emerged: intersection types are not only useful for extending simple types but also for *refining* them [4]. One thus obtains finer information on simply typed terms by assigning intersection types. This in particular leads to the concept of *normalization by typing*, that turns out to be quite efficient in some situations [5]. Moreover, intersection types are invariant under β-equivalence, so that they constitute a denotational semantics in a natural way [1]. Finally, intersection types also work in an infinitary setting, where terms may represent infinite trees and types play the role of automata. This leads to a model checking framework for higher order recursion schemes via intersection types [2, 3].

The purpose of this talk is to outline the recent development of intersection types described above. In particular, we explain how an efficient evaluation algorithm is obtained by combining normalization by typing, β-reduction and Krivine's abstract machine, to result in the following complexity characterization. Consider simply typed lambda terms of boolean type $o \to o \to o$ and of order r. Then the problem of deciding whether a given term evaluates to "true" is complete for n-EXPTIME if $r = 2n + 2$, and complete for n-EXPSPACE if $r = 2n + 3$ [5].

1998 ACM Subject Classification F.4.1 Mathematical Logic

Keywords and phrases simply typed lambda calculus, computational complexity, denotational semantics, intersection types

Digital Object Identifier 10.4230/LIPIcs.FSTTCS.2013.41

Category Invited Talk

References

1 Thomas Ehrhard. Collapsing non-idempotent intersection types. In *Proceedings of 26th CSL*, LIPIcs, Vol. 16, pages 259–273, 2012.
2 Naoki Kobayashi. Types and higher-order recursion schemes for verification of higher-order programs. In *Proceedings of 36th POPL*, pages 416–428, 2009.
3 Naoki Kobayashi and C.-H. Luke Ong. A type system equivalent to the modal mu-calculus model checking of higher-order recursion schemes. In *Proceedings of 24th LICS*, pages 179–188, 2009.
4 Sylvain Salvati. On the membership problem for non-linear abstract categorial grammars. *Journal of Logic, Language and Information*, 19(2):163–183, 2010.

5 Kazushige Terui. Semantic evaluation, intersection types and complexity of simply typed lambda calculus. In *Proceedings of 23rd RTA*, LIPIcs, Vol. 15, pages 323–338, 2012.

6 Steffen van Bakel. Intersection type assignment systems. *Theoretical Computer Science*, 151(2):385–435, 1995.