Invited Talk

Subtyping and Intersection Types Revisited

Frank Pfenning
Department of Computer Science
Carnegie Mellon University
fp@cs.cmu.edu

Abstract

Church's system of simple types has proven to be remarkably robust: call-by-name, call-by-need, and call-by-value languages, with or without effects, and even logical frameworks can be based on the same typing rules. When type systems become more expressive, this unity fractures. An early example is the value restriction for parametric polymorphism which is necessary for ML but not Haskell; a later manifestation is the lack of distributivity of function types over intersections in call-by-value languages with effects.

In this talk we reexamine the logical justification for systems of subtyping and intersection types and then explore the consequences in two different settings: logical frameworks and functional programming.

In logical frameworks functions are pure and their definitions observable, but complications could arise from the presence of dependent types. We show that this is not the case, and that we can obtain soundness and completeness theorems for a certain axiomatization of subtyping. We also sketch a connection to the type-theoretic notion of proof irrelevance.

In functional programming we investigate how the encapsulation of effects in monads interacts with subtyping and intersection types, providing an updated analysis of the value restriction and related phenomena (Davies and Pfenning 2000). While at present this study is far from complete, we believe that its origin in purely logical notions will give rise to a uniform theory that can easily be adapted to specific languages and their operational interpretations.

Categories and Subject Descriptors D.3.1 [*Programming Languages*]: Formal Definitions and Theory; F.3.3 [*Logics and Meanings of Programs*]: Studies of Program Constructs—Type structure; F.4.1 [*Mathematical Logic and Formal Languages*]: Mathematical Logic

General Terms Languages, Theory, Verification

Acknowledgments

The material in this invited talk is based on joint work with Rowan Davies (Pfenning and Davies 2001), William Lovas (Lovas and Pfenning 2007), and Noam Zeilberger (Zeilberger 2007).

Bio

Frank Pfenning received his Ph.D. in Mathematics in 1987 from Carnegie Mellon University and subsequently joined the Department of Computer Science at CMU as research faculty where he became Professor in 2002 and Director of Graduate Programs in 2004. He has spent time as visiting scientist at the Max-Planck-Institute for Computer Science in Saarbrücken, as Alexander-von-Humboldt Fellow at the Technical University Darmstadt, and as Visiting Professor at École Polytechnique and INRIA-Futurs. He has advised 19 completed Ph.D. theses and won the Herbert A. Simon Award for Teaching Excellence in the School of Computer Science in 2002. He has written extensive lecture notes on a variety of topics, including computation and deduction, constructive logic, linear logic, automated theorem proving, and logic programming. He served as trustee, vice president, and president of CADE, Inc., the governing body of the International Conference on Automated Deduction and on advisory boards for INRIA, the Max-Planck-Institute for Computer Science, and the School of Computer Science at Seoul National University. He has chaired the PPDP and CADE conferences and several program committees, including GPCE, RTA, CADE, and LICS.

His research interests include programming languages, logic and type theory, logical frameworks, automated deduction, and, most recently, logical methods in security. He has contributed to the development of refinement types and dependent types, the operational interpretation of modal λ-calculi for staged and distributed computation, and the theory and practice of higher-order logic programming. He has been a codeveloper of the Twelf meta-logical framework, which has been used for a number of formalization and verification efforts in programming languages, and participated in the design of its linear (LLF) and concurrent (CLF) extensions.

References

Rowan Davies and Frank Pfenning. Intersection types and computational effects. In P. Wadler, editor, *Proceedings of the Fifth International Conference on Functional Programming (ICFP'00)*, pages 198–208, Montreal, Canada, September 2000. ACM Press.

William Lovas and Frank Pfenning. A bidirectional refinement type system for LF. In B. Pientka and C. Schürmann, editors, *Proceedings of the Second International Workshop on Logical Frameworks and Meta-Languages: Theory and Practice*, pages 11–25, Bremen, Germany, July 2007.

Frank Pfenning and Rowan Davies. A judgmental reconstruction of modal logic. *Mathematical Structures in Computer Science*, 11:511–540, 2001.

Noam Zeilberger. On the unity of duality. *Annals of Pure and Applied Logic*, 2007. To appear in a special issue on *Classical Logic and Computation*.