Type Refinements for the Working Class

Jon Sterling and Darin Morrison

There are two conflicting views which bedevil any discussion of the nature of type theory. First, there is the notion of type theory as an extension or generalization of universal algebra to support interdependency of sorts and operations, possibly subject to an arbitrary equational theory [3, 6]; we will call this *formal type theory*. Typing, in such a setting, is a mere matter of grammar and is nearly always decidable. In hindsight, we may observe that this is the sort of type theory which Martin-Löf first proposed in 1972 [18], even if we will admit that this was not the intention at the time. A model for such a type theory is usually given by interpreting the types or sorts as presheaves or sheaves over contexts of hypotheses, and as such, a proof theoretic interpretation of the hypothetical judgment is inevitable.

Secondly, there is the view of type theory as semi-formal theory of constructions for the Brouwer-Heyting-Kolmogorov interpretation of intuitionistic mathematical language, which we will call *behavioral* or *semantic type theory*. The most widely known development of this program is Martin-Löf's 1979 "extensional" type theory [17, 19], but we must give priority to Dana Scott for inventing this line of research in 1970 with his prophetic report, *Constructive Validity* [21]. More recently, behavioral type theory has been developed much further in the Nuprl family [4] of proof assistants, including MetaPRL [14] and JonPRL [22].

Martin-Löf's key innovation was the commitment to pervasive functionality (extensionality) as part of the *definitions* of the judgments and the types, in contrast to the state of affairs in formal type theory where functionality is a metatheorem which may or may not hold depending on the particular equational theory which has been imposed. Furthermore, models for behavioral type theory interpret the types as partial equivalence relations on only closed terms, and the meaning of the hypothetical judgment is defined separately and uniformly in the logical relations style.

Our position is that these views of type theory are not in conflict, but rather merely describe two distinct layers in a single, harmonious system. From this perspective, formal type theories can do little more than negotiate matters of grammar, and therefore may serve as a syntactic (linguistic) framework for mathematical language, being responsible for the management of variable binding and substitution. On the other hand, the meaning of mathematical statements shall be specified *behaviorally* in the semantic type theory.

The types of the semantic theory can then be said to *refine* the types of the syntactic theory, both by placing restrictions on membership and by coarsening equivalence. Thus far, all developments of behavioral type theory have been built on a *unityped* syntactic framework, and so the relation to type refinements has been difficult to see. In this paper, we contribute a full theory of behavioral

refinements over multi-sorted abstract binding trees, a simple formal type theory [12, 23]; this hybrid system allows the deployment of a Nuprl-style type theory over any signature of sorts and operators.

References

[1] P. Aczel. A general Church-Rosser theorem. Technical report, University of Manchester, 1978.

[2] T. Altenkirch, J. Chapman, and T. Uustalu. Monads need not be endofunctors. *Logical Methods in Computer Science*, 11(1:3):1–40, 2015.

[3] J. Cartmell. Generalised algebraic theories and contextual categories. *Annals of Pure and Applied Logic*, 32:209 – 243, 1986.

[4] R. L. Constable, S. F. Allen, H. M. Bromley, W. R. Cleaveland, J. F. Cremer, R. W. Harper, D. J. Howe, T. B. Knoblock, N. P. Mendler, P. Panangaden, J. T. Sasaki, and S. F. Smith. *Implementing Mathematics with the Nuprl Proof Development System*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1986.

[5] K. Crary. Explicit contexts in LF (extended abstract). *Electronic Notes in Theoretical Computer Science*, 228:53 – 68, 2009. Proceedings of the International Workshop on Logical Frameworks and Metalanguages: Theory and Practice (LFMTP 2008).

[6] P. Dybjer. Internal type theory. In S. Berardi and M. Coppo, editors, *Types for Proofs and Programs*, volume 1158 of *Lecture Notes in Computer Science*, pages 120–134. Springer Berlin Heidelberg, 1996.

[7] M. Fiore and C.-K. Hur. Second-order equational logic (extended abstract). In A. Dawar and H. Veith, editors, *Computer Science Logic*, volume 6247 of *Lecture Notes in Computer Science*, pages 320–335. Springer Berlin Heidelberg, 2010.

[8] M. Fiore and O. Mamoud. Second-order algebraic theories – (extended abstract). In *Mathematical Foundations of Computer Science 2010, 35th International Symposium, MFCS 2010, Brno, Czech Republic, August 23-27, 2010. Proceedings*, pages 368–380, 2010.

[9] M. Fiore, G. Plotkin, and D. Turi. Abstract syntax and variable binding. In *Proceedings of the 14th Symposium on Logic in Computer Science*, pages 193–202, 1999.

[10] M. P. Fiore. Mathematical models of computational and combinatorial structures. In *Foundations of Software Science and Computational Structures, 8th International Conference, FOSSACS 2005, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2005, Edinburgh, UK, April 4-8, 2005, Proceedings*, pages 25–46, 2005.

[11] M. Hamana. Free Σ-monoids: A higher-order syntax with metavariables. In W.-N. Chin, editor, *Programming Languages and Systems*, volume 3302 of *Lecture Notes in Computer Science*, pages 348–363. Springer Berlin Heidelberg, 2004.

[12] R. Harper. *Practical Foundations for Programming Languages*. Cambridge University Press, New York, NY, USA, 2016.

[13] R. Harper, F. Honsell, and G. Plotkin. A framework for defining logics. *J. ACM*, 40(1):143–184, Jan. 1993.

[14] J. Hickey, A. Nogin, R. L. Constable, B. E. Aydemir, E. Barzilay, Y. Bryukhov, R. Eaton, A. Granicz, A. Kopylov, C. Kreitz, V. N. Krupski, L. Lorigo, S. Schmitt, C. Witty, and X. Yu. MetaPRL – a modular logical environment. In D. Basin and B. Wolff, editors, *Theorem Proving in Higher Order Logics*, volume 2758 of *Lecture Notes in Computer Science*, pages 287–303. Springer Berlin Heidelberg, 2003.

[15] D. K. Lee, K. Crary, and R. Harper. Towards a mechanized metatheory of Standard ML. In *Proceedings of the 34th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, POPL '07, pages 173–184, New York, NY, USA, 2007. ACM.

[16] C. Martens and K. Crary. LF in LF: Mechanizing the metatheories of LF in Twelf. In *Proceedings of the Seventh International Workshop on Logical Frameworks and Meta-languages, Theory and Practice*, LFMTP '12, pages 23–32, New York, NY, USA, 2012. ACM.

[17] P. Martin-Löf. Constructive mathematics and computer programming. In L. J. Cohen, J. Łoś, H. Pfeiffer, and K.-P. Podewski, editors, *Logic, Methodology and Philosophy of Science VI, Proceedings of the Sixth International Congress of Logic, Methodology and Philosophy of Science, Hannover 1979*, volume 104 of *Studies in Logic and the Foundations of Mathematics*, pages 153–175. North-Holland, 1982.

[18] P. Martin-Löf. An intuitionistic theory of types, 1972.

[19] P. Martin-Löf and G. Sambin. *Intuitionistic type theory*. Studies in proof theory. Bibliopolis, Napoli, 1984.

[20] U. Norell. *Towards a practical programming language based on dependent type theory*. PhD thesis, Department of Computer Science and Engineering, Chalmers University of Technology, SE-412 96 Göteborg, Sweden, September 2007.

[21] D. Scott. Constructive validity. In M. Laudet, D. Lacombe, L. Nolin, and M. Schützenberger, editors, *Symposium on Automatic Demonstration*, volume 125 of *Lecture Notes in Mathematics*, pages 237–275. Springer Berlin Heidelberg, 1970.

[22] J. Sterling, D. Gratzer, and V. Rahli. JonPRL. `http://www.jonprl.org/`, 2015.

[23] J. Sterling and D. Morrison. Syntax and Semantics of Abstract Binding Trees. 2015.