Nominal Sets and Dependent Type Theory

Andrew M. Pitts

Computer Laboratory
University of Cambridge
Cambridge CB3 0FD, UK

Nominal sets [3, 7] provide a mathematical theory of structures involving names and binding constructs, based on some simple, but subtle ideas going back to Fraenkel and Mostowski's symmetric models of set theory with atoms. The theory has been applied to programming language semantics, machine-assisted theorem proving and the design of functional and logical metaprogramming languages. In this talk I want to explore the relationship between nominal sets and dependent type theory, with the following two motivations in mind, both of which involve the nominal sets notion of *name abstraction*.

Homotopy Type Theory. The cubical sets model of homotopy type theory was introduced by Bezem, Coquand and Huber [1] using a category of presheaves. This category is equivalent to a category of nominal sets equipped operations for substituting contants 0 and 1 for names (the names in this case being names of cartesian axes $x, y, z, \ldots$); see [6]. In the nominal version of the model, proofs of identity are given by name abstractions: abstracting a named direction x in an element a gives a path (proof of equality) from $a[0/x]$ to $a[1/x]$. In order to interpret dependent types, the category of nominal sets can be extended to a category with families [2, 4] in a straightforward way.

Constructive nominal logic. FreshML [8] adds name abstraction types to ML [5], allowing the user to declare inductively defined data involving name binding operations and define functions on such data using patterns involving bound names. The semantics of FreshML guarantees that programmers cannot break α-conversion, while allowing them to use a style close to informal practice when manipulating structures with bound names. I would very much like to have a similarly usable language that completes the following proportion:

$$\frac{\text{Agda}}{\text{Haskell}} = \frac{?}{\text{FreshML}}$$

Achieving this convincingly requires versions of the nominal sets notions of *freshness*, *name abstraction* and *name restriction* within constructive type theory that have good meta-theoretic properties and yet are syntactically simple from a user's point of view.

References

[1] M. Bezem, T. Coquand, and S. Huber. A model of type theory in cubical sets. Preprint, September 2013.

[2] Peter 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.

[3] M. J. Gabbay. Foundations of nominal techniques: Logic and semantics of variables in abstract syntax. *Bulletin of Symbolic Logic*, 17(2):161–229, 2011.

[4] M. Hofmann. Syntax and semantics of dependent types. In A. M. Pitts and P. Dybjer, editors, *Semantics and Logics of Computation*, Publications of the Newton Institute, pages 79–130. Cambridge University Press, 1997.

[5] R. Milner, M. Tofte, R. Harper, and D. MacQueen. *The Definition of Standard ML (Revised)*. MIT Press, 1997.

[6] A. M. Pitts. An equivalent presentation of the Bezem-Coquand-Huber category of cubical sets. Preprint arXiv:1401.7807 [cs.LO], December 2013.

[7] A. M. Pitts. *Nominal Sets: Names and Symmetry in Computer Science*, volume 57 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 2013.

[8] M. R. Shinwell, A. M. Pitts, and M. J. Gabbay. FreshML: Programming with binders made simple. In *Eighth ACM SIGPLAN International Conference on Functional Programming (ICFP 2003), Uppsala, Sweden*, pages 263–274. ACM Press, August 2003.