Mathematical Structures in Dependent Type Theory

Assia Mahboubi

Inria, Nantes, France
LS2N, Université de Nantes, France
Vrije Universiteit Amsterdam, The Netherlands
`http://people.rennes.inria.fr/Assia.Mahboubi/`
Assia.Mahboubi@inria.fr

Abstract

In this talk, we discuss the role and the implementation of mathematical structures in libraries of formalised mathematics in dependent type theory.

2012 ACM Subject Classification Theory of computation → Type theory; Theory of computation → Logic and verification

Keywords and phrases Mathematical structures, formalized mathematics, dependent type theory

Digital Object Identifier 10.4230/LIPIcs.CSL.2021.2

Category Invited Talk

1 Summary

Since the early writings of the Nicolas Bourbaki group [3], *mathematical structures* are used in all fields of mathematics to structure the mathematical language, its vocabulary and its notational apparatus. An instance of a given structure is a carrier set equipped with some identified elements, with some operations on the carrier, and with some properties – called the axioms of the structure. Put in good use, these abstractions clarify the mathematical discourse for a knowledgeable audience, while emphasising correspondences between seemingly unrelated mathematical objects. Classical model theory provides a mathematical formalisation of the notion of *structure* [12], of which algebraic structures are an instance.

The past decade has seen the advent of several large-scale libraries of formalised mathematics [6, 2, 9, 17], most of which framed by a hierarchy of formal algebraic structures [10, 13, 8, 17]. The latter hierarchies can be seen as a formal-proof-engineering device, which organises inheritance and sharing in a similar way as the design patterns of object-oriented programming [7, 4]. The implementation and the features of these hierarchies depend both on the flavour of foundations the proof assistant is based on, and on the implementation in the prover of enhanced type inference procedures [15, 11, 16, 1, 14]. The central idea is to take benefit of some form of type inference in order to compute automatically the missing information in the user input, so as to achieve concision in the statement of formal sentences, while still providing a well-formed term to the prover's checker.

This talk will focus more specifically on the case of formalisations, and proof assistants, based on variants of dependent type theory. This setting allows in particular a first-class representation of structures using dependent tuples (also called *telescopes* [5]). It will discuss the recent techniques proposed to design these hierarchies, their pitfalls, the corresponding achievements, and their limitations.

References

1 Andrea Asperti, Wilmer Ricciotti, Claudio Sacerdoti Coen, and Enrico Tassi. Hints in unification. In Stefan Berghofer, Tobias Nipkow, Christian Urban, and Makarius Wenzel,

© Assia Mahboubi;
licensed under Creative Commons License CC-BY

editors, *Theorem Proving in Higher Order Logics, 22nd International Conference, TPHOLs 2009, Munich, Germany, August 17-20, 2009. Proceedings*, volume 5674 of *Lecture Notes in Computer Science*, pages 84–98. Springer, 2009. doi:10.1007/978-3-642-03359-9_8.

2 Grzegorz Bancerek, Czeslaw Bylinski, Adam Grabowski, Artur Kornilowicz, Roman Matuszewski, Adam Naumowicz, and Karol Pak. The role of the mizar mathematical library for interactive proof development in mizar. *J. Autom. Reason.*, 61(1-4):9–32, 2018. doi:10.1007/s10817-017-9440-6.

3 Nicholas Bourbaki. The Architecture of Mathematics. *The American Mathematical Monthly*, 57(4), 1950.

4 Cyril Cohen, Kazuhiko Sakaguchi, and Enrico Tassi. Hierarchy builder: Algebraic hierarchies made easy in coq with elpi (system description). In Zena M. Ariola, editor, *5th International Conference on Formal Structures for Computation and Deduction, FSCD 2020, June 29-July 6, 2020, Paris, France (Virtual Conference)*, volume 167 of *LIPIcs*, pages 34:1–34:21. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.FSCD.2020.34.

5 N. G. de Bruijn. Telescopic mappings in typed lambda calculus. *Information and Computation*, 91(2):189–204, 1991.

6 Manuel Eberl, Gerwin Klein, Tobias Nipkow, Larry Paulson, and René Thiemann. Archive of Formal Proofs. https://www.isa-afp.org/about.html.

7 Erich Gamma, Richard Helm, Ralph Johnson, and John M. Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley Professional, 1st edition, 1994.

8 François Garillot, Georges Gonthier, Assia Mahboubi, and Laurence Rideau. Packaging mathematical structures. In *Theorem Proving in Higher-Order Logics (TPHOL 2009)*, volume 5674 of *Lecture Notes in Computer Science*, pages 327–342. Springer, 2009.

9 Georges Gonthier, Andrea Asperti, Jeremy Avigad, Yves Bertot, Cyril Cohen, François Garillot, Stéphane Le Roux, Assia Mahboubi, Russell O'Connor, Sidi Ould Biha, Ioana Pasca, Laurence Rideau, Alexey Solovyev, Enrico Tassi, and Laurent Théry. A machine-checked proof of the odd order theorem. In *4th International Conference on Interactive Theorem Proving (ITP 2013), Rennes, France, July 22–26, 2013*, volume 7998 of *Lecture Notes in Computer Science*, pages 163–179. Springer, 2013.

10 Adam Grabowski, Artur Kornilowicz, and Christoph Schwarzweller. On algebraic hierarchies in mathematical repository of Mizar. In Maria Ganzha, Leszek A. Maciaszek, and Marcin Paprzycki, editors, *Proceedings of the 2016 Federated Conference on Computer Science and Information Systems, FedCSIS 2016, Gdańsk, Poland, September 11-14, 2016*, volume 8 of *Annals of Computer Science and Information Systems*, pages 363–371. IEEE, 2016. doi:10.15439/2016F520.

11 Florian Haftmann and Makarius Wenzel. Constructive type classes in isabelle. In Thorsten Altenkirch and Conor McBride, editors, *Types for Proofs and Programs, International Workshop, TYPES 2006, Nottingham, UK, April 18-21, 2006, Revised Selected Papers*, volume 4502 of *Lecture Notes in Computer Science*, pages 160–174. Springer, 2006. doi:10.1007/978-3-540-74464-1_11.

12 Wilfrid Hodges. *Model Theory*. Encyclopedia of Mathematics and its Applications. Cambridge University Press, 1993. doi:10.1017/CBO9780511551574.

13 Johannes Hölzl, Fabian Immler, and Brian Huffman. Type classes and filters for mathematical analysis in Isabelle/HOL. In *4th International Conference on Interactive Theorem Proving (ITP 2013), Rennes, France, July 22–26, 2013*, volume 7998 of *Lecture Notes in Computer Science*, pages 279–294. Springer, 2013.

14 Assia Mahboubi and Enrico Tassi. Canonical structures for the working Coq user. In *4th International Conference on Interactive Theorem Proving (ITP 2013), Rennes, France, July 22–26, 2013*, volume 7998 of *Lecture Notes in Computer Science*, pages 19–34. Springer, 2013.

15 Amokrane Saïbi. Typing algorithm in type theory with inheritance. In Peter Lee, Fritz Henglein, and Neil D. Jones, editors, *Conference Record of POPL'97: The 24th ACM SIGPLAN-*

SIGACT Symposium on Principles of Programming Languages, Papers Presented at the Symposium, Paris, France, 15-17 January 1997, pages 292–301. ACM Press, 1997. `doi:10.1145/263699.263742`.

16 Matthieu Sozeau and Nicolas Oury. First-Class Type Classes. In *21st International Conference on Theorem Proving in Higher Order Logics (TPHOLs 2008), Montréal, Québec, Canada, August 18–21, 2008*, volume 5170 of *Lecture Notes in Computer Science*, pages 278–293. Springer, 2008.

17 The mathlib Community. The Lean mathematical library. In *9th ACM SIGPLAN International Conference on Certified Programs and Proofs (CPP 2020), New Orleans, LA, USA, January 20–21, 2020*, pages 367–381. ACM, 2020.