Hammering Higher Order Set Theory

Chad E. Brown[1], Cezary Kaliszyk[2], Martin Suda[1], and Josef Urban[1]

[1] Czech Technical University in Prague
{martin.suda,josef.urban}@gmail.com
[2] Univesity of Melbourne and University of Innsbruck
ckaliszyk@unimelb.edu.au

Abstract

We use automated theorem provers (ATPs) combined with premise selection to shorten proofs in the Megalodon library. As the proof foundation is higher-order set theory, it closely aligns with the logic of higher-order ATPs making such tools effective. Furthermore, many of the subgoals are even first-order which allows the use of even more effective first-order ATPs. We evaluate several provers and discuss proof reconstruction for ATP-derived proofs.

1 Megalodon Library Overview

Megalodon is a minimal interactive theorem prover. The underlying framework is simply-typed intuitionistic higher-order logic with Curry-Howard proof terms, using a single base type ι for sets, a built-in type o for propositions, and function types $\alpha \to \beta$. The framework only includes implication and universal quantification, but is expressive enough to support impredicative encodings of logical connectives and quantifiers. On top of the framework there are set-theoretic primitives and axioms, including Grothendieck universes and an ε-choice operator. In this work, we describe a comprehensive formal development in Megalodon that culminates in formal proofs of 12 theorems from the Freek 100 list – a widely recognized benchmark suite for interactive theorem provers. The development begins with foundational logic and set-theoretic definitions, including ordinals and natural numbers as finite ordinals. A central feature of the library is a novel set-theoretic encoding of *Conway's surreal numbers* extending the ordinals and facilitate recursive definitions, including definitions of the usual field operations as in [5]. Once the ordered field of surreal numbers is constructed, one can carve out the set of real numbers and the set of integers. This provides the material to state and prove theorems such as Bezout's Theorem, the greatest common divisor algorithm, the Fundamental Theorem of Arithmetic, and the irrationality of $\sqrt{2}$. The entire development, given as a single 45,000-line file with 139 definitions and 999 theorems, is presented with highly detailed proof scripts with minimal automation.

2 Automating with ATPs

We extended Megalodon by the tactic "`aby`" that generates TH0 problems for higher-order ATPs and FOF problems for first-order ones. As the foundation is already higher-order logic, the translation to TH0 TPTP is straightforward. In addition, if the problem (dependencies and conjecture) is in the first-order fragment, a FOF problem can also be created directly. In order to identify subproofs that could be replaced by calls to ATPs, we used Megalodon to generate the problem files for most tactic calls, determining the dependencies when the subproof is completed.

Vampire (s/h)	Vampire (ho)	Zipper-position	E	Lash	cvc5
32675 (78.3%)	32474 (77.8%)	31310 (75%)	23866 (57.2%)	14987 (35.9%)	13238 (31.7%)

Table 1: Higher order ATPs on premise-selected ("bushy") subgoals

We benchmarked several ATPs including Vampire [3], Zipperposition [2], E [9], Lash [4], and cvc5 [1]. Of the 41738 subgoals, 82% could be solved by some ATP in 60 seconds (Fig. 1). The resulting new ATP benchmark is available online[1].

Megalodon additionally produced 29880 first order problem files when the subgoal had a first order proof. Among the 11179 problems that could be solved by Vampire only 11 were not already among the ones solved by any higher-order prover.

The automatically found proofs are often shorter than the original ones. If we restrict to the largest texts that could be replaced (along with some manual modifications), we were left with 3401 calls to an ATP and a development with 17435 lines and 159363 characters. This is roughly 46% the original size of the development. The majority of the proofs in the development could be replaced simply by a single `aby` call.

3 Hammer Integration

We also implemented a simple Emacs mode for Megalodon[2]. It allows one to insert an `aby` call after an ATP finds a proof, automatically extracting used dependencies. To test the performance of the full hammer we evaluated the 51243 subgoals[3] using all previous facts (the so called "chainy" mode). Vampire solved 59.5% of these within 60 seconds. Future work includes using ML-based premise selection methods, which is likely to improve this performance further, as well as the addition of ML guiding methods into higher-order ATPs/SMTs such as Vampire, E, Lash and cvc5.

4 Proof Reconstruction

Megalodon currently lacks built-in automation that would allow for internal proof reconstruction. Even if the ATPs can verify the subgoals externally, building a Megalodon proof term is challenging. Prover9 [8] includes an IVY export, that can be used for first-order proof reconstruction; similarly a recent version of Vampire [7] includes support for exporting proofs (in Dedukti [6] format) that are easier to check. Full integration of these tools remains future work.

References

[1] Haniel Barbosa, Clark W. Barrett, Martin Brain, Gereon Kremer, Hanna Lachnitt, Makai Mann, Abdalrhman Mohamed, Mudathir Mohamed, Aina Niemetz, Andres Nötzli, Alex Ozdemir, Mathias Preiner, Andrew Reynolds, Ying Sheng, Cesare Tinelli, and Yoni Zohar. cvc5: A versatile and industrial-strength SMT solver. In Dana Fisman and Grigore Rosu, editors, *Tools and Algorithms for the Construction and Analysis of Systems - 28th International Conference, TACAS 2022, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2022, Munich,*

[1] https://github.com/MgUser36/MegalodonATPBenchmark
[2] https://github.com/MgUser36/megalodon/blob/main/mg-advice.el
[3] https://github.com/MgUser36/MegalodonATPBenchmark/tree/main/chainy

Germany, April 2-7, 2022, Proceedings, Part I, volume 13243 of *Lecture Notes in Computer Science*, pages 415–442. Springer, 2022.

[2] Alexander Bentkamp, Jasmin Blanchette, Simon Cruanes, and Uwe Waldmann. Superposition for lambda-free higher-order logic. *Log. Methods Comput. Sci.*, 17(2), 2021.

[3] Ahmed Bhayat and Martin Suda. A higher-order vampire (short paper). In Christoph Benzmüller, Marijn J. H. Heule, and Renate A. Schmidt, editors, *Automated Reasoning - 12th International Joint Conference, IJCAR 2024, Nancy, France, July 3-6, 2024, Proceedings, Part I*, volume 14739 of *Lecture Notes in Computer Science*, pages 75–85. Springer, 2024.

[4] Chad E. Brown and Cezary Kaliszyk. Lash 1.0 (system description). In Jasmin Blanchette, Laura Kovács, and Dirk Pattinson, editors, *Automated Reasoning - 11th International Joint Conference, IJCAR 2022, Haifa, Israel, August 8-10, 2022, Proceedings*, volume 13385 of *Lecture Notes in Computer Science*, pages 350–358. Springer, 2022.

[5] John H. Conway. *On numbers and games, Second Edition.* A K Peters, 2001.

[6] The Dedukti logical framework. https://deducteam.github.io.

[7] Anja Petković Komel, Michael Rawson, and Martin Suda. Case study: Verified vampire proofs in the lambdapi-calculus modulo, 2025.

[8] W. McCune. Prover9 and mace4. http://www.cs.unm.edu/~mccune/prover9/, 2005–2010.

[9] Petar Vukmirović, Jasmin Blanchette, and Stephan Schulz. Extending a high-performance prover to higher-order logic. In Sriram Sankaranarayanan and Natasha Sharygina, editors, *Tools and Algorithms for the Construction and Analysis of Systems - 29th International Conference, TACAS 2023, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2022, Paris, France, April 22-27, 2023, Proceedings, Part II*, volume 13994 of *Lecture Notes in Computer Science*, pages 111–129. Springer, 2023.