McTT: Building A Correct-By-Construction Proof Checker For Martin-Löf Type Theory

Junyoung Jang
junyoung.jang@mail.mcgill.ca
McGill University
Montréal, Québec, Canada

Jason Z. S. Hu
zhong.s.hu@mail.mcgill.ca
McGill University
Montréal, Québec, Canada

Antoine Gaulin
antoine.gaulin@mail.mcgill.ca
McGill University
Montréal, Québec, Canada

Brigitte Pientka
brigitte.pientka@mcgill.ca
McGill University
Montréal, Québec, Canada

ACM Reference Format:
Junyoung Jang, Jason Z. S. Hu, Antoine Gaulin, and Brigitte Pientka. 2024. McTT: Building A Correct-By-Construction Proof Checker For Martin-Löf Type Theory. In *Proceedings of ACM Conference (Conference'17)*. ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

Over the past decades, proof assistants based on type theories have been widely successful from verifying safety-critical software to establishing a new standard of rigour by formalizing mathematics. But proof assistants are also complex pieces of software, and software invariably has bugs, so why should we trust such a proof assistant?

Proof assistants are typically designed with an eye toward minimizing such concerns, relying on a small, trusted core kernel to construct and verify a proof. This design approach focuses the concern to the trusted core, which is supposed to be close to the underlying theory, i.e. the Calculus of Inductive Constructions (CIC) [Coquand and Paulin 1988; Paulin-Mohring 1993] and Martin-Löf Type Theory (MLTT) [Martin-Löf 1984; Martin-Löf 1975] that often has been extensively studied. However, in practice, kernels of modern proof assistants are no longer small enough from a human perspective and they significantly deviate and extend the core type theory that has been investigated in theory.

Popular proof assistants like Coq (soon to be known as Rocq) [The Coq Development Team 2023], Agda [The Agda Team 2024] and Lean [de Moura and Ullrich 2021; de Moura et al. 2015] include type-checking kernels spanning over tens of thousands of lines of code. Furthermore, extensions that are often added to the kernel include meta-variables and inductive type families. From usability and engineering perspectives, error messaging become important. As a result, all three proof assistants have encountered major bugs: In Coq, on average, one critical bug has been found every year and Lean has experienced both soundness bugs and segmentation faults.

Conference'17, July 2017, Washington, DC, USA
2024. ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

We propose a correct-by-construction way to develop proof assistants and describe McTT, a certified implementation of Martin-Löf type theory (MLTT) in Coq. McTT is composed of two major parts: the theoretical component and the executable component. In the theoretical component, we formalize a version of MLTT that has natural numbers, propositional equality types, Π types and a full cumulative universe hierarchy. In addition, we support covariant universe subtyping which subsumes types from lower universes to higher ones and propagates under the output types of Π types. In order to model the full universe hierarchy, we adapt Hu et al. [2023]'s mechanization based on induction-recursion in Agda to an impredicative encoding of the semantics with subtyping in Coq's `Prop` universe. Using these semantics, we prove the completeness and soundness of the normalization-by-evaluation algorithm (NbE) using an untyped domain model [Abel 2013] and conclude the logical consistency of the type theory as a corollary. Compared to previous similar mechanization efforts [Adjedj et al. 2024; Wieczorek and Biernacki 2018], McTT formalizes a richer type theory.

In addition to the theoretical component, McTT has an executable component. Here, we relate the model of normalization in the theoretical component and the actual implementation with equivalences, so that we can extract a correct-by-construction type checker in OCaml and couple it with a parser front-end for execution. The resulting executable type-checker can then be used to type-check MLTT specifications written in a source file.

The extracted code is of high quality; it is identical to what a skilled human programmer would have written, and thus the resulting executable is also efficient.

McTT provides the a basic framework for certified implementations of dependent type theories and is open to many possible future extensions. Therefore, McTT the first step for future explorations in this direction. The source code and a homepage is available on Github [Jang et al. 2024]. Everyone is welcome to pull our repository and try out our project by following the instructions in the README file. Instructions are also available on the homepage.

References

Andreas Abel. 2013. *Normalization by Evaluation: Dependent Types and Impredicativity*. Habilitation Thesis. Ludwig-Maximilians-Universität München, Munich, Germany. https://www.cse.chalmers.se/~abela/habil.pdf

Arthur Adjedj, Meven Lennon-Bertrand, Kenji Maillard, Pierre-Marie Pédrot, and Loïc Pujet. 2024. Martin-Löf à la Coq. In *Proceedings of the 13th ACM SIGPLAN International Conference on Certified Programs and Proofs, CPP 2024, London, UK, January 15-16, 2024*, Amin Timany, Dmitriy Traytel, Brigitte Pientka, and Sandrine Blazy (Eds.). ACM, 230–245. https://doi.org/10.1145/3636501.3636951

Thierry Coquand and Christine Paulin. 1988. Inductively Defined Types. In *COLOG-88, International Conference on Computer Logic, Tallinn, USSR, December 1988, Proceedings (Lecture Notes in Computer Science)*, Per Martin-Löf and Grigori Mints (Eds.), Vol. 417. Springer, 50–66. https://doi.org/10.1007/3-540-52335-9_47

Leonardo de Moura and Sebastian Ullrich. 2021. The Lean 4 Theorem Prover and Programming Language. In *Proceedings of the 28th International Conference on Automated Deduction, CADE 2021, Virtual Event, July 12-15, 2021 (Lecture Notes in Computer Science)*, André Platzer and Geoff Sutcliffe (Eds.), Vol. 12699. Springer, 625–635. https://doi.org/10.1007/978-3-030-79876-5_37

Leonardo Mendonça de Moura, Soonho Kong, Jeremy Avigad, Floris van Doorn, and Jakob von Raumer. 2015. The Lean Theorem Prover (System Description). In *Proceedings of the 25th International Conference on Automated Deduction, CADE 2015, Berlin, Germany, August 1-7, 2015 (Lecture Notes in Computer Science)*, Amy P. Felty and Aart Middeldorp (Eds.), Vol. 9195. Springer, 378–388. https://doi.org/10.1007/978-3-319-21401-6_26

Jason Z. S. Hu, Junyoung Jang, and Brigitte Pientka. 2023. Normalization by Evaluation for Modal Dependent Type Theory. *J. Funct. Program.* 33 (2023). https://doi.org/10.1017/S0956796823000060

Junyoung Jang, Jason Z. S. Hu, Antoine Gaulin, and Brigitte Pientka. 2024. McTT: A Bottom-up Approach to Implementing A Proof Assistant. https://beluga-lang.github.io/McTT/

Per Martin-Löf. 1984. *Intuitionistic Type Theory*. Studies in proof theory, Vol. 1. Bibliopolis.

Per Martin-Löf. 1975. An Intuitionistic Theory of Types: Predicative Part. In *Logic Colloquium '73*, H.E. Rose and J.C. Shepherdson (Eds.). Studies in Logic and the Foundations of Mathematics, Vol. 80. Elsevier, 73–118. https://doi.org/10.1016/S0049-237X(08)71945-1

Christine Paulin-Mohring. 1993. Inductive Definitions in the System Coq - Rules and Properties. In *Typed Lambda Calculi and Applications, International Conference on Typed Lambda Calculi and Applications, TLCA '93, Utrecht, the Netherlands, March 16-18, 1993, Proceedings (Lecture Notes in Computer Science)*, Marc Bezem and Jan Friso Groote (Eds.), Vol. 664. Springer, 328–345. https://doi.org/10.1007/BFB0037116

The Agda Team. 2024. Agda 2.6.4.3. https://wiki.portal.chalmers.se/agda/pmwiki.php

The Coq Development Team. 2023. *The Coq Proof Assistant.* https://doi.org/10.5281/zenodo.8161141

Pawel Wieczorek and Dariusz Biernacki. 2018. A Coq Formalization of Normalization by Evaluation for Martin-Löf Type Theory. In *Proceedings of the 7th ACM SIGPLAN International Conference on Certified Programs and Proofs, CPP 2018, Los Angeles, California, USA, January 8-9, 2018*, June Andronick and Amy P. Felty (Eds.). ACM, 266–279. https://doi.org/10.1145/3167091