QED at Large: A Survey of Engineering of Formally Verified Software

Other titles in Foundations and Trends® in Programming Languages

Tutorial on Static Inference of Numeric Invariants by Abstract Interpretation
Antoine Miné
ISBN: 978-1-68083-386-7

Program Synthesis
Sumit Gulwani, Oleksandr Polozov and Rishabh Singh
ISBN: 978-1-68083-292-1

Programming with Big Code
Martin Vechev and Eran Yahav
ISBN: 978-1-68083-230-3

Behavioral Types in Programming Languages
Davide Ancona et al.
ISBN: 978-1-68083-134-4

Computer-Assisted Query Formulation
Alvin Cheung and Armando Solar-Lezama
ISBN: 978-1-68083-036-1

Full text available at: http://dx.doi.org/10.1561/2500000045

QED at Large: A Survey of Engineering of Formally Verified Software

Talia Ringer
University of Washington
tringer@cs.washington.edu

Karl Palmskog
University of Texas at Austin
palmskog@acm.org

Ilya Sergey
Yale-NUS College
and
National University of Singapore
ilya.sergey@yale-nus.edu.sg

Milos Gligoric
University of Texas at Austin
gligoric@utexas.edu

Zachary Tatlock
University of Washington
ztatlock@cs.washington.edu

now
the essence of knowledge
Boston — Delft

Full text available at: http://dx.doi.org/10.1561/2500000045

Foundations and Trends® in Programming Languages

Published, sold and distributed by:
now Publishers Inc.
PO Box 1024
Hanover, MA 02339
United States
Tel. +1-781-985-4510
www.nowpublishers.com
sales@nowpublishers.com

Outside North America:
now Publishers Inc.
PO Box 179
2600 AD Delft
The Netherlands
Tel. +31-6-51115274

The preferred citation for this publication is

T. Ringer, K. Palmskog, I. Sergey, M. Gligoric, Z. Tatlock. *QED at Large: A Survey of Engineering of Formally Verified Software*. Foundations and Trends® in Programming Languages, vol. 5, no. 2-3, pp. 102–281, 2019.

ISBN: 978-1-68083-595-3
© 2019 T. Ringer, K. Palmskog, I. Sergey, M. Gligoric, Z. Tatlock

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, mechanical, photocopying, recording or otherwise, without prior written permission of the publishers.

Photocopying. In the USA: This journal is registered at the Copyright Clearance Center, Inc., 222 Rosewood Drive, Danvers, MA 01923. Authorization to photocopy items for internal or personal use, or the internal or personal use of specific clients, is granted by now Publishers Inc for users registered with the Copyright Clearance Center (CCC). The 'services' for users can be found on the internet at: www.copyright.com

For those organizations that have been granted a photocopy license, a separate system of payment has been arranged. Authorization does not extend to other kinds of copying, such as that for general distribution, for advertising or promotional purposes, for creating new collective works, or for resale. In the rest of the world: Permission to photocopy must be obtained from the copyright owner. Please apply to now Publishers Inc., PO Box 1024, Hanover, MA 02339, USA; Tel. +1 781 871 0245; www.nowpublishers.com; sales@nowpublishers.com

now Publishers Inc. has an exclusive license to publish this material worldwide. Permission to use this content must be obtained from the copyright license holder. Please apply to now Publishers, PO Box 179, 2600 AD Delft, The Netherlands, www.nowpublishers.com; e-mail: sales@nowpublishers.com

Full text available at: http://dx.doi.org/10.1561/2500000045

Foundations and Trends® in Programming Languages

Volume 5, Issue 2-3, 2019

Editorial Board

Editor-in-Chief

Mooly Sagiv
Tel-Aviv University, Israel

Editors

Martín Abadi
Google and UC Santa Cruz

Anindya Banerjeei
IMDEA Software Instituet

Patrick Cousoti
ENS, Paris and NYU

Oege De Moori
University of Oxford

Matthias Felleiseni
Northeastern University

John Fieldi
Google

Cormac Flanagani
UC Santa Cruz

Philippa Gardneri
Imperial College

Andrew Gordoni
Microsoft Research and University of Edinburgh

Dan Grossmani
University of Washington

Robert Harper
CMU

Tim Harris
Amazon

Fritz Hengleini
University of Copenhagen

Rupak Majumdari
MPI and UCLA

Kenneth McMillani
Microsoft Research

J. Eliot B. Mossi
University of Massachusetts, Amherst

Andrew C. Myersi
Cornell University

Hanne Riis Nielsoni
Technical University of Denmark

Peter O'Hearni
University College London

Benjamin C. Piercei
University of Pennsylvania

Andrew Pittsi
University of Cambridge

Ganesan Ramalingami
Microsoft Research

Mooly Sagivi
Tel Aviv University

Davide Sangiorgii
University of Bologna

David Schmidti
Kansas State University

Peter Sewelli
University of Cambridge

Scott Stolleri
Stony Brook University

Peter Stuckeyi
University of Melbourne

Jan Viteki
Northeastern University

Philip Wadleri
University of Edinburgh

David Walkeri
Princeton University

Stephanie Weirici
University of Pennsylvania

Editorial Scope

Topics

Foundations and Trends® in Programming Languages publishes survey and tutorial articles in the following topics:

- Abstract Interpretation
- Compilation and Interpretation Techniques
- Domain Specific Languages
- Formal Semantics, including Lambda Calculi, Process Calculi, and Process Algebra
- Language Paradigms
- Mechanical Proof Checking
- Memory Management
- Partial Evaluation
- Program Logic
- Programming Language Implementation
- Programming Language Security
- Programming Languages for Concurrency
- Programming Languages for Parallelism
- Program Synthesis
- Program Transformations and Optimizations
- Program Verification
- Runtime Techniques for Programming Languages
- Software Model Checking
- Static and Dynamic Program Analysis
- Type Theory and Type Systems

Information for Librarians

Foundations and Trends® in Programming Languages, 2019, Volume 5, 4 issues. ISSN paper version 2325-1107. ISSN online version 2325-1131. Also available as a combined paper and online subscription.

Contents

QED at Large: A Survey of Engineering of Formally Verified Software

Talia Ringer[1], Karl Palmskog[2], Ilya Sergey[3], Milos Gligoric[4] and Zachary Tatlock[5]

[1] *University of Washington; tringer@cs.washington.edu*
[2] *University of Texas at Austin; palmskog@utexas.edu*
[3] *Yale-NUS College; ilya.sergey@yale-nus.edu.sg*
[4] *University of Texas at Austin; gligoric@utexas.edu*
[5] *University of Washington; ztatlock@cs.washington.edu*

ABSTRACT

Development of formal proofs of correctness of programs can increase actual and perceived reliability and facilitate better understanding of program specifications and their underlying assumptions. Tools supporting such development have been available for over 40 years, but have only recently seen wide practical use. Projects based on construction of machine-checked formal proofs are now reaching an unprecedented scale, comparable to large software projects, which leads to new challenges in proof development and maintenance. Despite its increasing importance, the field of proof engineering is seldom considered in its own right; related theories, techniques, and tools span many fields and venues. This survey of the literature presents a holistic understanding of proof engineering for program correctness, covering impact in practice, foundations, proof automation, proof organization, and practical proof development.

Talia Ringer, Karl Palmskog, Ilya Sergey, Milos Gligoric and Zachary Tatlock (2019), "QED at Large: A Survey of Engineering of Formally Verified Software", Foundations and Trends® in Programming Languages: Vol. 5, No. 2-3, pp 102–281. DOI: 10.1561/2500000045.

1

Introduction

A formal proof of program correctness can show that for all possible inputs, the program behaves as expected. This theoretical guarantee can provide practical benefits. For example, the formally verified optimizing C compiler CompCert (Leroy, 2006) is empirically more reliable than GCC and LLVM: the test generation tool Csmith (Yang *et al.*, 2011) found 79 bugs in GCC and 202 bugs in LLVM, but was unable to find any bugs in the verified parts of CompCert.

Methodologies for developing proofs of program correctness are as old as the proofs themselves (Turing, 1949; Floyd, 1967; Hoare, 1971). These proofs were on paper and of simple programs; tools to support their development followed soon after (Milner, 1972) and have continued to evolve for over 40 years (Bjørner and Havelund, 2014). Projects based on construction of formal, machine-checked proofs using these tools are now reaching a scale comparable to that of large software engineering projects. For example, the initial correctness proofs for an operating system kernel took around 20 person years to develop (Klein *et al.*, 2009), and as of 2014 consisted of 480,000 lines of specifications and proof scripts (Klein *et al.*, 2014).

This survey covers the timeline and research literature concerning proof development for program verification, including theories, languages, and tools. It emphasizes challenges and breakthroughs at each stage in history and highlights challenges that are currently present due to the increasing scale of proof developments.

1.1 Challenges at Scale

Scaling up leads to new challenges and additional demand for tool support in proof development and maintenance. For example, users may have to reformulate properties to facilitate library reuse (Hales *et al.*, 2017), or to encode data structures in specific ways to aid in automation of proofs about them (Gonthier, 2008). Proof development environments need to allow users to efficiently write, check, and share proofs (Faithfull *et al.*, 2018); proof libraries need to allow easy search and seamless integration of results into local developments (Gauthier and Kaliszyk, 2015). Evolving projects face the possibility of previous proofs breaking due to seemingly unrelated changes, justifying design principles (Woos *et al.*, 2016) as well as support for quick error detection (Celik *et al.*, 2017) and repair (Ringer *et al.*, 2018).

The research community has answered these challenges with theories, techniques, and tools for proofs of program correctness that scale—all of which fall under the umbrella of *proof engineering*, or software engineering for proofs. Many of these techniques draw inspiration from work in software engineering on large-scale development practices and tools (Klein, 2014). However, even with close conceptual ties between construction of programs and proofs, research in software engineering requires careful translation to the world of formal proofs. For example, proof engineers can benefit from regression testing techniques by considering lemmas and their proofs in place of tests, as in *regression proving* (Celik *et al.*, 2017); yet, the standard metric used to prioritize regression tests—statement coverage—has no clear analogue for lemmas with complex conditions and quantification.

This survey serves to gather these theories, techniques, and tools into a single place, drawing parallels to software engineering, and pointing out challenges that are especially pronounced in proof development.

It discusses the problems engineers encounter when verifying large systems, existing solutions to these problems, and future opportunities for research to address underserved problems.

1.2 Scope: Domain and Literature

We consider proof engineering research in the context of interactive theorem provers (ITPs) or *proof assistants* (used interchangeably with ITPs in this survey) that satisfy the *de Bruijn criterion* (Barendregt and Barendsen, 2002; Barendregt and Wiedijk, 2005), which requires that they produce proof objects that a small proof-checking kernel can verify; the general workflow of such tools is illustrated in Figure 1.1. That is, we consider proof assistants such as Coq (Coq Development Team, 1989-2019), Isabelle/HOL (Isabelle Development Team, 1994-2019), HOL Light (HOL Light Development Team, 1996-2019), and Agda (Agda Development Team, 2007-2019); we do not consider program verifiers, theorem provers, and constraint solvers such as Dafny (Leino, 2010), ACL2 (ACL2 Development Team, 1990-2019), and Z3 (Z3 Development Team, 2008-2019) except when contributions carry over. We focus on proof engineering for software verification, but consider contributions from mathematics and other domains when relevant.

Sometimes, the key design principles in *engineering* a large program verification effort are not the focus of the most well-known publications on the effort. Instead, they can be in less standard references such as workshop papers (Komendantskaya *et al.*, 2012; Paulson and Blanchette, 2012; Pit-Claudel and Courtieu, 2016; Mulhern, 2006), invited talks (Wenzel, 2017a), blog posts (*Verified cryptography for Firefox 57* n.d.), and online documents (Leroy, 2017; Wenzel, 2017b). One purpose of this survey is to bring such design principles front-and-center. Naturally, we shall aim to survey the relevant literature with our best effort to provide accurate and thorough citations. To that end, we will not hesitate to cite both traditional research papers in well-known venues and relevant discussions in less traditional forms, without further distinction among them.

1.3 Overview

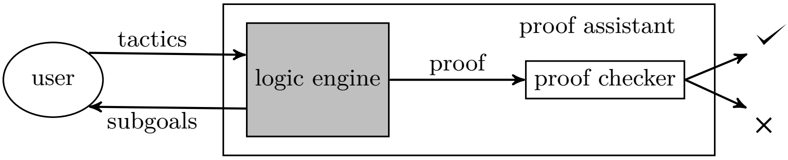

Figure 1.1: Typical proof assistant workflow, adapted from Geuvers (2009).

After motivating chapters (Chapters 2 and 3), this survey discusses the history and foundations of proof assistants (Chapter 4). It then surveys proof engineering research under three headings: languages and automation (Chapter 5), proof organization and scalability (Chapter 6), and practical proof development and evolution (Chapter 7). At a glance, Chapter 5 concerns proof automation approaches and languages, Chapter 6 concerns methods to express and organize programs and proofs, and Chapter 7 concerns development processes and tools. Each of these three chapters is divided into sections; each section surveys a more granular area of proof engineering research, then concludes with a discussion of opportunities for future work within that area when applicable. The survey concludes (Chapter 8) with a discussion of opportunities for future work within proof engineering more broadly. In the case of factual errors, an errata may be found on https://proofengineering.org.

1.4 Reading Guide

This survey aims to reach a broad audience of researchers, proof engineers, and community members who are interested in understanding, using, or contributing to proof engineering research. Readers need not be deterred for lack of background knowledge. It is not always necessary to understand previous chapters in order to understand later chapters; readers should feel free to skip sections or chapters, or to consult later chapters or cited resources for more information. This guide lists topics with which basic familiarity is helpful in order to get the most out of the referenced chapters (all chapters unless otherwise specified), along with resources (cited next to ⌧ icons) for the interested reader:

- Programming languages, type systems, and metatheory ⌧ (Pierce, 2002; Harper, 2016), including:
    - ITPs ⌧ (Geuvers, 2009; Harrison *et al.*, 2014), especially:
        - Coq ⌧ (Chlipala, 2013a; Pierce *et al.*, 2014; Bertot and Casteran, 2004)
        - Isabelle/HOL ⌧ (Wenzel *et al.*, 2004; Nipkow and Klein, 2014)
    - Automated reasoning ⌧ (Bradley and Manna, 2007; Kroening and Strichman, 2008) (Chapters 3 and 5)
    - The Curry-Howard correspondence ⌧ (Pfenning, 2010; Sørensen and Urzyczyn, 2006)
    - Dependent and inductive types ⌧ (Chlipala, 2013a)
    - Equality ⌧ (Chlipala, 2013a; nLab authors, 2019a) (Chapters 3 and 4)
    - Compilers ⌧ (Cooper and Torczon, 2011) (Chapters 3, 4, and 6)
- Software engineering ⌧ (Shaw *et al.*, 2015)
- Systems ⌧ (Anderson and Dahlin, 2014; Cachin *et al.*, 2011) (Chapters 3 and 6)
- Formalized mathematics ⌧ (American Mathematical Society, 2008; nLab authors, 2019b) (Chapter 3)

References

Abel, A., A. Momigliano, and B. Pientka. 2017a. “POPLMark Reloaded”. In: *Proceedings of the Logical Frameworks and Meta-Languages: Theory and Practice Workshop*.

Abel, A., A. Vezzosi, and T. Winterhalter. 2017b. “Normalization by Evaluation for Sized Dependent Types”. *Proc. ACM Program. Lang.* 1(ICFP): 33:1–33:30. DOI: 10.1145/3110277.

ACL2 Development Team. 1990-2019. “ACL2”. URL: http://www.cs.utexas.edu/users/moore/acl2.

ACM SIGPLAN. 2016. “Most Influential POPL Paper Award”. URL: http://www.sigplan.org/Awards/POPL.

Adams, M. 2015. “Refactoring Proofs with Tactician”. In: *Software Engineering and Formal Methods*. Berlin, Heidelberg: Springer. 53–67. DOI: 10.1007/978-3-662-49224-6_6.

Agda Development Team. 2005-2017. “Quick Guide to Editing, Type Checking and Compiling Agda Code – Agda 2.5.4.2 Documentation”. URL: http://agda.readthedocs.io/en/v2.5.4.2/getting-started/quick-guide.html#quick-guide-introduction.

Agda Development Team. 2005-2018. “Cubical Type Theory in Agda – Agda 2.6.0 Documentation”. URL: http://agda.readthedocs.io/en/latest/language/cubical.html.

Agda Development Team. 2007-2019. “The Agda Wiki”. URL: http://wiki.portal.chalmers.se/agda/pmwiki.php.

Aitken, J., P. Gray, T. Melham, and M. Thomas. 1998. “Interactive Theorem Proving: An Empirical Study of User Activity”. *Journal of Symbolic Computation*. 25(2): 263–284. DOI: 10.1006/jsco.1997.0175.

Aitken, S. and T. Melham. 2000. “An analysis of errors in interactive proof attempts”. *Interacting with Computers*. 12(6): 565–586. DOI: 10.1016/S0953-5438(99)00023-5.

Aldrich, J., R. J. Simmons, and K. Shin. 2008. “SASyLF: An Educational Proof Assistant for Language Theory”. In: *Proceedings of the 2008 International Workshop on Functional and Declarative Programming in Education. FDPE ’08*. Victoria, BC, Canada: ACM. 31–40. DOI: 10.1145/1411260.1411266.

Allen, S. F., R. L. Constable, D. J. Howe, and W. E. Aitken. 1990. “The semantics of reflected proof”. In: *Proceedings. Fifth Annual IEEE Symposium on Logic in Computer Science*. 95–105. DOI: 10.1109/LICS.1990.113737.

Altenkirch, T., V. Gaspes, B. Nordström, and B. von Sydow. 1994. “A user’s guide to ALF”. URL: http://www.cse.chalmers.se/~bengt/papers/usersguidetoalf.pdf.

Amani, S., A. Hixon, Z. Chen, C. Rizkallah, P. Chubb, L. O’Connor, J. Beeren, Y. Nagashima, J. Lim, T. Sewell, J. Tuong, G. Keller, T. Murray, G. Klein, and G. Heiser. 2016. “Cogent: Verifying High-Assurance File System Implementations”. In: *International Conference on Architectural Support for Programming Languages and Operating Systems*. Atlanta, GA, USA. 175–188. DOI: 10.1145/2872362.2872404.

Ambler, S. J., R. L. Crole, and A. Momigliano. 2002. “Combining Higher Order Abstract Syntax with Tactical Theorem Proving and (Co)Induction”. In: *Theorem Proving in Higher Order Logics*. Berlin, Heidelberg: Springer. 13–30. DOI: 10.1007/3-540-45685-6_3.

Notices of the American Mathematical Society. 2008: *A Special Issue on Formal Proof*. URL: http://www.ams.org/notices/200811/.

Amin, N. and T. Rompf. 2017. “Type Soundness Proofs with Definitional Interpreters”. In: *Proceedings of the 44th ACM SIGPLAN Symposium on Principles of Programming Languages. POPL 2017*. Paris, France: ACM. 666–679. DOI: 10.1145/3009837.3009866.

Anand, A., A. W. Appel, G. Morrisett, M. Weaver, M. Sozeau, O. Savary Belanger, R. Pollack, and Z. Paraskevopoulou. 2017. "CertiCoq: A verified compiler for Coq". In: *CoqPL*. URL: http://www.cs.princeton.edu/~appel/papers/certicoq-coqpl.pdf.

Anand, A., S. Boulier, C. Cohen, M. Sozeau, and N. Tabareau. 2018. "Towards Certified Meta-Programming with Typed Template-Coq". In: *Interactive Theorem Proving*. Cham: Springer International Publishing. 20–39. DOI: 10.1007/978-3-319-94821-8_2.

Anand, A. and V. Rahli. 2014. "Towards a Formally Verified Proof Assistant". In: *Interactive Theorem Proving*. Cham: Springer International Publishing. 27–44. DOI: 10.1007/978-3-319-08970-6_3.

Anderson, T. and M. Dahlin. 2014. *Operating Systems: Principles and Practice*. 2nd. Recursive books.

Andronick, J., R. Jeffery, G. Klein, R. Kolanski, M. Staples, H. Zhang, and L. Zhu. 2012. "Large-Scale Formal Verification in Practice: A Process Perspective". In: *International Conference on Software Engineering*. Zurich, Switzerland: ACM. 1002–1011. DOI: 10.1109/ICSE.2012.6227120.

Angiuli, C., E. Cavallo, K. Hou, R. Harper, and J. Sterling. 2018. "The RedPRL Proof Assistant (Invited Paper)". In: *Proceedings of the 13th International Workshop on Logical Frameworks and Meta-Languages: Theory and Practice, LFMTP@FSCD 2018, Oxford, UK, 7th July 2018*. 1–10. DOI: 10.4204/EPTCS.274.1.

Appel, A. W. 2006. "Tactics for separation logic". *INRIA Rocquencourt and Princeton University, Early Draft*. URL: https://www.cs.princeton.edu/~appel/papers/septacs.pdf.

Appel, A. W. 2017. Personal communication.

Appel, A. W. 2011a. "Efficient Verified Red-Black Trees". URL: https://www.cs.princeton.edu/~appel/papers/redblack.pdf.

Appel, A. W. 2011b. "Verified Software Toolchain". In: *Programming Languages and Systems: 20th European Symposium on Programming, ESOP 2011, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2011, Saarbrücken, Germany, March 26–April 3, 2011. Proceedings*. Berlin, Heidelberg: Springer. 1–17. DOI: 10.1007/978-3-642-19718-5_1.

Appel, A. W., R. Dockins, A. Hobor, L. Beringer, J. Dodds, G. Stewart, S. Blazy, and X. Leroy. 2014. *Program Logics for Certified Compilers*. Cambridge University Press. DOI: 10.1017/CBO9781107256552.

Appel, A. W., N. G. Michael, A. Stump, and R. Virga. 2003. "A Trustworthy Proof Checker". *J. Autom. Reasoning*. 31(3-4): 231–260. DOI: 10.1023/B:JARS.0000021013.61329.58.

Appel, A. W., R. Dockins, and X. Leroy. 2012. "A list-machine benchmark for mechanized metatheory". *Journal of Automated Reasoning*. 49(3): 453–491. DOI: 10.1007/s10817-011-9226-1.

Aristotle. 1926. "Prior Analytics". In: *The Works of Aristotle*. Translated by A. J. Jenkinson. Oxford.

Armstrong, A., T. Bauereiss, B. Campbell, A. Reid, K. E. Gray, R. M. Norton, P. Mundkur, M. Wassell, J. French, C. Pulte, S. Flur, I. Stark, N. Krishnaswami, and P. Sewell. 2019. "ISA Semantics for ARMv8-a, RISC-v, and CHERI-MIPS". *Proc. ACM Program. Lang.* 3(POPL): 71:1–71:31. DOI: 10.1145/3290384.

Asperti, A., C. S. Coen, E. Tassi, and S. Zacchiroli. 2007. "User interaction with the Matita proof assistant". *Journal of Automated Reasoning*. 39(2): 109–139. DOI: 10.1007/s10817-007-9070-5.

Asperti, A., F. Guidi, C. S. Coen, E. Tassi, and S. Zacchiroli. 2004. "A content based mathematical search engine: Whelp". In: *International Workshop on Types for Proofs and Programs*. Springer. 17–32. DOI: 10.1007/11617990_2.

Aspinall, D. 2000a. "Isamode". URL: https://homepages.inf.ed.ac.uk/da/Isamode/doc/Isamode_toc.html.

Aspinall, D. 2000b. "Proof General: A Generic Tool for Proof Development". In: *Tools and Algorithms for the Construction and Analysis of Systems: 6th International Conference, TACAS 2000 Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2000 Berlin, Germany, March 25 – April 2, 2000 Proceedings*. Berlin, Heidelberg: Springer. 38–43. DOI: 10.1007/3-540-46419-0_3.

Aspinall, D. and A. Compagnoni. 2001. "Subtyping dependent types". *Theoretical Computer Science*. 266(1): 273–309. DOI: 10.1016/S0304-3975(00)00175-4.

Aspinall, D. and C. Kaliszyk. 2016a. "Towards Formal Proof Metrics". In: *Fundamental Approaches to Software Engineering*. Berlin, Heidelberg: Springer. 325–341. DOI: 10.1007/978-3-662-49665-7_19.

Aspinall, D. and C. Kaliszyk. 2016b. "What's in a Theorem Name?" In: *Interactive Theorem Proving*. Cham: Springer International Publishing. 459–465. DOI: 10.1007/978-3-319-43144-4_28.

Avigad, J. 2004. "Proof Mining". URL: http://www.andrew.cmu.edu/user/avigad/Talks/asl04.pdf.

Aydemir, B. E., A. Bohannon, M. Fairbairn, J. N. Foster, B. C. Pierce, P. Sewell, D. Vytiniotis, G. Washburn, S. Weirich, and S. Zdancewic. 2005. "Mechanized Metatheory for the Masses: The POPLMark Challenge". In: *Proceedings of the 18th International Conference on Theorem Proving in Higher Order Logics. TPHOLs'05*. Oxford, UK: Springer-Verlag. 50–65. DOI: 10.1007/11541868_4.

Aydemir, B. E., A. Bohannon, M. Fairbairn, J. N. Foster, B. C. Pierce, P. Sewell, D. Vytiniotis, G. Washburn, S. Weirich, and S. Zdancewic. 2005-2019. "The POPLMark Challenge". URL: http://www.seas.upenn.edu/~plclub/poplmark/.

Aydemir, B., A. Bohannon, and S. Weirich. 2006. "Nominal Reasoning Techniques in Coq". In: *International Workshop on Logical Frameworks and Meta-Languages:Theory and Practice (LFMTP)*. Seattle, WA, USA.

Aydemir, B., A. Charguéraud, B. C. Pierce, R. Pollack, and S. Weirich. 2008. "Engineering Formal Metatheory". In: *Proceedings of the 35th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages. POPL '08*. San Francisco, CA, USA: ACM. 3–15. DOI: 10.1145/1328438.1328443.

Back, R. J. R. 1991. "Refinement Diagrams". In: *4th Refinement Workshop*. London: Springer London. 125–137. DOI: 10.1007/978-1-4471-3756-6_7.

Back, R.-J. 1988. "A calculus of refinements for program derivations". *Acta Informatica*. 25(6): 593–624. DOI: 10.1007/BF00291051.

Backus, J. 1978. "Can Programming Be Liberated from the von Neumann Style?: A Functional Style and Its Algebra of Programs". *Commun. ACM*. 21(8): 613–641. DOI: 10.1145/359576.359579.

Bahr, P. and G. Hutton. 2015. "Calculating correct compilers". *J. Funct. Program.* 25. DOI: 10.1017/S0956796815000180.

Ballarin, C. 2006. "Interpretation of Locales in Isabelle: Theories and Proof Contexts". In: *Mathematical Knowledge Management*. Berlin, Heidelberg: Springer. 31–43. DOI: 10.1007/11812289_4.

Bancerek, G. 1990. "The fundamental properties of natural numbers". *Formalized Mathematics*. 1(1): 41–46.

Bansal, K., S. M. Loos, M. N. Rabe, C. Szegedy, and S. Wilcox. 2019. "HOList: An Environment for Machine Learning of Higher-Order Theorem Proving (extended version)". *CoRR*. abs/1904.03241.

Barendregt, H. 2007. "Proofs of Correctness in Mathematics and Industry". In: *Wiley Encyclopedia of Computer Science and Engineering*. John Wiley & Sons. DOI: 10.1002/9780470050118.ecse579.

Barendregt, H. 2013. "Foundations of Mathematics from the Perspective of Computer Verification". In: *Mathematics, Computer Science and Logic - A Never Ending Story: The Bruno Buchberger Festschrift*. Cham: Springer International Publishing. 1–49. DOI: 10.1007/978-3-319-00966-7_1.

Barendregt, H. and E. Barendsen. 2002. "Autarkic Computations in Formal Proofs". *Journal of Automated Reasoning*. 28(3): 321–336. DOI: 10.1023/A:1015761529444.

Barendregt, H. and F. Wiedijk. 2005. "The challenge of computer mathematics". *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*. 363(1835): 2351–2375. DOI: 10.1098/rsta.2005.1650.

Barras, B. 2010. "Sets in Coq, Coq in Sets". *Journal of Formalized Reasoning*. 3(1): 29–48. DOI: 10.6092/issn.1972-5787/1695.

Barras, B., L. del Carmen González Huesca, H. Herbelin, Y. Régis-Gianas, E. Tassi, M. Wenzel, and B. Wolff. 2013. "Pervasive Parallelism in Highly-Trustable Interactive Theorem Proving Systems". In: *Intelligent Computer Mathematics: MKM, Calculemus, DML, and Systems and Projects 2013, Held as Part of CICM 2013, Bath, UK, July 8-12, 2013. Proceedings*. Berlin, Heidelberg: Springer. 359–363. DOI: 10.1007/978-3-642-39320-4_29.

Barras, B., C. Tankink, and E. Tassi. 2015. "Asynchronous Processing of Coq Documents: From the Kernel up to the User Interface". In: *Interactive Theorem Proving: 6th International Conference, ITP 2015, Nanjing, China, August 24-27, 2015, Proceedings*. Cham: Springer International Publishing. 51–66. DOI: 10.1007/978-3-319-22102-1_4.

Barras, B. and B. Werner. 1997. "Coq in Coq". URL: http://www.lix.polytechnique.fr/Labo/Bruno.Barras/publi/coqincoq.pdf.

Barret, C. 2018. "Twitter Response". URL: http://twitter.com/cbarrett/status/1059881370260684824.

Barthe, G. 1995. "Implicit coercions in type systems". In: *International Workshop on Types for Proofs and Programs*. Springer. 1–15. DOI: 10.1007/3-540-61780-9_58.

Barthe, G. and P. Courtieu. 2002. "Efficient reasoning about executable specifications in Coq". In: *International Conference on Theorem Proving in Higher Order Logics*. Springer. 31–46. DOI: 10.1007/3-540-45685-6_4.

Barthe, G. and O. Pons. 2001. "Type Isomorphisms and Proof Reuse in Dependent Type Theory". In: *Foundations of Software Science and Computation Structures*. Berlin, Heidelberg: Springer. 57–71. DOI: 10.1007/3-540-45315-6_4.

Barthe, G., M. Ruys, and H. Barendregt. 1996. "A two-level approach towards lean proof-checking". In: *Types for Proofs and Programs: International Workshop, TYPES '95 Torino, Italy, June 5–8, 1995 Selected Papers*. Berlin, Heidelberg: Springer. 16–35. DOI: 10.1007/3-540-61780-9_59.

Bates, J. L. 1979. "A Logic for Correct Program Development". AAI8003896. *PhD thesis*. Ithaca, NY, USA: Cornell University.

Bauer, A., J. Gross, P. L. Lumsdaine, M. Shulman, M. Sozeau, and B. Spitters. 2017. "The HoTT Library: A Formalization of Homotopy Type Theory in Coq". In: *Proceedings of the 6th ACM SIGPLAN Conference on Certified Programs and Proofs. CPP 2017*. Paris, France: ACM. 164–172. DOI: 10.1145/3018610.3018615.

Beckert, B., S. Grebing, and F. Böhl. 2014. "A Usability Evaluation of Interactive Theorem Provers Using Focus Groups". In: *12th International Conference on Software Engineering and Formal Methods (SEFM 2014) – Collocated Workshops: Human-Oriented Formal Methods (HOFM 2014)*. Vol. 8938. *Lecture Notes in Computer Science*. Springer. 3–19. DOI: 10.1007/978-3-319-15201-1_1.

Benthem Jutting, L. S. van. 1994. "Checking Landau's Grundlagen in the Automath System: Parts of Chapters 0, 1 and 2 (Introduction, Prepration, Translation)". In: *Studies in Logic and the Foundations of Mathematics*. Vol. 133. Elsevier. 701–720. DOI: 10.1016/S0049-237X(08)70222-2.

Berger, U., S. Berghofer, P. Letouzey, and H. Schwichtenberg. 2005. "Program extraction from normalization proofs". *Studia Logica*. 82. DOI: 10.1007/s11225-006-6604-5.

Berghofer, S. and T. Nipkow. 2002. "Executing Higher Order Logic". In: *Types for Proofs and Programs: International Workshop, TYPES 2000 Durham, UK, December 8–12, 2000 Selected Papers*. Berlin, Heidelberg: Springer. 24–40. DOI: 10.1007/3-540-45842-5_2.

Berghofer, S. and C. Urban. 2007. "A Head-to-Head Comparison of de Bruijn Indices and Names". *Electronic Notes in Theoretical Computer Science*. 174(5): 53–67. DOI: 10.1016/j.entcs.2007.01.018.

Berghofer, S. and M. Wenzel. 1999. "Inductive Datatypes in HOL — Lessons Learned in Formal-Logic Engineering". In: *Theorem Proving in Higher Order Logics*. Berlin, Heidelberg: Springer. 19–36. DOI: 10.1007/3-540-48256-3_3.

Bertot, Y. 2009. "Theorem proving support in programming language semantics". In: *From Semantics to Computer Science: Essays in Honour of Gilles Kahn*. Ed. by Y. Bertot, G. Huet, J.-J. Lévy, and G. Plotkin. Cambridge University Press. 337–361. URL: http://hal.inria.fr/inria-00160309/.

Bertot, Y. and P. Casteran. 2004. *Interactive Theorem Proving and Program Development: Coq'Art: The Calculus of Inductive Constructions. Texts in Theoretical Computer Science An EATCS Series.* Berlin, Heidelberg: Springer. DOI: 10.1007/978-3-662-07964-5.

Bertot, Y., G. Kahn, and L. Théry. 1994. "Proof by pointing". In: *Theoretical Aspects of Computer Software*. Berlin, Heidelberg: Springer. 141–160. DOI: 10.1007/3-540-57887-0_94.

Bertot, Y. and L. Théry. 1998. "A Generic Approach to Building User Interfaces for Theorem Provers". *Journal of Symbolic Computation*. 25(2): 161–194. DOI: 10.1006/jsco.1997.0171.

Biendarra, J., J. C. Blanchette, A. Bouzy, M. Desharnais, M. Fleury, J. Hölzl, O. Kunčar, A. Lochbihler, F. Meier, L. Panny, A. Popescu, C. Sternagel, R. Thiemann, and D. Traytel. 2017. "Foundational (Co)datatypes and (Co)recursion for Higher-Order Logic". In: *Frontiers of Combining Systems*. Cham: Springer International Publishing. 3–21. DOI: 10.1007/978-3-319-66167-4_1.

Birkedal, L. and A. Bizjak. 2018. "Lecture Notes on Iris: Higher-Order Concurrent Separation Logic". URL: https://iris-project.org/tutorial-pdfs/iris-lecture-notes.pdf.

Bishop, E. and D. Bridges. 1985. *Constructive Analysis*. Springer. DOI: 10.1007/978-3-642-61667-9.

Bishop, S., M. Fairbairn, H. Mehnert, M. Norrish, T. Ridge, P. Sewell, M. Smith, and K. Wansbrough. 2018. "Engineering with Logic: Rigorous Test-Oracle Specification and Validation for TCP/IP and the Sockets API". *J. ACM*. 66(1): 1:1–1:77. DOI: 10.1145/3243650.

Bishop, S., M. Fairbairn, M. Norrish, P. Sewell, M. Smith, and K. Wansbrough. 2006. "Engineering with Logic: HOL Specification and Symbolic-evaluation Testing for TCP Implementations". In: *Conference Record of the 33rd ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages. POPL '06*. Charleston, SC, USA: ACM. 55–66. DOI: 10.1145/1111037.1111043.

Bjørner, D. and K. Havelund. 2014. "40 Years of Formal Methods". In: *FM 2014: Formal Methods: 19th International Symposium, Singapore, May 12-16, 2014. Proceedings*. Cham: Springer International Publishing. 42–61. DOI: 10.1007/978-3-319-06410-9_4.

Blanchette, J. C., L. Bulwahn, and T. Nipkow. 2011. "Automatic Proof and Disproof in Isabelle/HOL". In: *Frontiers of Combining Systems*. Berlin, Heidelberg: Springer. 12–27. DOI: 10.1007/978-3-642-24364-6_2.

Blanchette, J. C., M. Fleury, P. Lammich, and C. Weidenbach. 2018. "A verified SAT solver framework with learn, forget, restart, and incrementality". *Journal of Automated Reasoning*. 61(1-4): 333–365.

Blanchette, J. C., D. Greenaway, C. Kaliszyk, D. Kühlwein, and J. Urban. 2016a. "A Learning-Based Fact Selector for Isabelle/HOL". *Journal of Automated Reasoning*. 57(3): 219–244. DOI: 10.1007/s10817-016-9362-8.

Blanchette, J. C., M. Haslbeck, D. Matichuk, and T. Nipkow. 2015. "Mining the Archive of Formal Proofs". In: *Intelligent Computer Mathematics: International Conference, CICM 2015, Washington, DC, USA, July 13-17, 2015, Proceedings*. Cham: Springer International Publishing. 3–17. DOI: 10.1007/978-3-319-20615-8_1.

Blanchette, J. C., C. Kaliszyk, L. C. Paulson, and J. Urban. 2016b. "Hammering towards QED". *Journal of Formalized Reasoning*. 9(1): 101–148. DOI: 10.6092/issn.1972-5787/4593.

Blanchette, J. C. and T. Nipkow. 2010. "Nitpick: A Counterexample Generator for Higher-order Logic Based on a Relational Model Finder". In: *Proceedings of the First International Conference on Interactive Theorem Proving. ITP'10*. Edinburgh, UK: Springer-Verlag. 131–146. DOI: 10.1007/978-3-642-14052-5_11.

Boite, O. 2004. "Proof Reuse with Extended Inductive Types". In: *Theorem Proving in Higher Order Logics: 17th International Conference, TPHOLs 2004, Park City, Utah, USA, September 14-17, 2004. Proceedings*. Berlin, Heidelberg: Springer. 50–65. DOI: 10.1007/978-3-540-30142-4_4.

Boulton, R. J., A. Gordon, M. J. C. Gordon, J. Harrison, J. Herbert, and J. V. Tassel. 1992. "Experience with Embedding Hardware Description Languages in HOL". In: *Proceedings of the IFIP TC10/WG 10.2 International Conference on Theorem Provers in Circuit Design: Theory, Practice and Experience*. Amsterdam, The Netherlands: North-Holland Publishing Co. 129–156.

Bounov, D., A. DeRossi, M. Menarini, W. G. Griswold, and S. Lerner. 2018. "Inferring Loop Invariants through Gamification". In: *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. ACM. 231. DOI: 10.1145/3173574.3173805.

Bourke, T., M. Daum, G. Klein, and R. Kolanski. 2012. "Challenges and Experiences in Managing Large-Scale Proofs". In: *Conferences on Intelligent Computer Mathematics (CICM) / Mathematical Knowledge Management*. Bremen, Germany: Springer. 32–48. DOI: 10.1007/978-3-642-31374-5_3.

Boutin, S. 1997. "Using reflection to build efficient and certified decision procedures". In: *Theoretical Aspects of Computer Software: Third International Symposium, TACS'97 Sendai, Japan, September 23–26, 1997 Proceedings*. Berlin, Heidelberg: Springer. 515–529. DOI: 10.1007/BFb0014565.

Boyer, R. S. 1994. "A mechanically proof-checked encyclopedia of mathematics: Should we build one? Can we?" In: *Automated Deduction — CADE-12: 12th International Conference on Automated Deduction Nancy, France, June 26 – July 1, 1994 Proceedings*. Berlin, Heidelberg: Springer. 237–251. DOI: 10.1007/3-540-58156-1_17.

Bradley, A. R. and Z. Manna. 2007. *The Calculus of Computation: Decision Procedures with Applications to Verification*. Berlin, Heidelberg: Springer-Verlag.

Brady, E. 2013. "Idris, a general-purpose dependently typed programming language: Design and implementation". *Journal of Functional Programming*. 23(5): 552–593. DOI: 10.1017/S095679681300018X.

Braibant, T. and D. Pous. 2011. "Tactics for Reasoning Modulo AC in Coq". In: *Certified Programs and Proofs: First International Conference, CPP 2011, Kenting, Taiwan, December 7-9, 2011. Proceedings*. Berlin, Heidelberg: Springer. 167–182. DOI: 10.1007/978-3-642-25379-9_14.

Braibant, T. and D. Pous. 2012. "Deciding Kleene Algebras in Coq". *Logical Methods in Computer Science*. Volume 8, Issue 1. DOI: 10.2168/LMCS-8(1:16)2012.

Buchberger, B. 2000. "Theory exploration with Theorema". *Analele Universitatii Din Timisoara, ser. Matematica-Informatica*. 38(2): 9–32.

Buchberger, B., A. Craciun, T. Jebelean, L. Kovács, T. Kutsia, K. Nakagawa, F. Piroi, N. Popov, J. Robu, M. Rosenkranz, *et al.* 2006. "Theorema: Towards computer-aided mathematical theory exploration". *Journal of Applied Logic.* 4(4): 470–504. DOI: 10.1016/j.jal.2005.10.006.

Bulwahn, L. 2012. "The New Quickcheck for Isabelle: Random, Exhaustive and Symbolic Testing Under One Roof". In: *Proceedings of the Second International Conference on Certified Programs and Proofs. CPP'12.* Kyoto, Japan: Springer-Verlag. 92–108. DOI: 10.1007/978-3-642-35308-6_10.

Bundy, A. 1988. "The Use of Explicit Plans to Guide Inductive Proofs". In: *Proceedings of the 9th International Conference on Automated Deduction.* Berlin, Heidelberg: Springer. 111–120. DOI: 10.1007/BFb0012826.

Burstall, R. M. 1969. "Proving Properties of Programs by Structural Induction". *The Computer Journal.* 12(1): 41–48. DOI: 10.1093/comjnl/12.1.41.

Busch, H. 1994. "First-order automation for higher-order-logic theorem proving". In: *Higher Order Logic Theorem Proving and Its Applications.* Springer. 97–112. DOI: 10.1007/3-540-58450-1_37.

Cachin, C., R. Guerraoui, and L. Rodrigues. 2011. *Introduction to Reliable and Secure Distributed Programming.* 2nd. Springer. DOI: 10.1007/978-3-642-15260-3.

Callaghan, P. and Z. Luo. 2001. "An implementation of LF with coercive subtyping & universes". *Journal of Automated Reasoning.* 27(1): 3–27. DOI: 10.1023/A:1010648911114.

Cao, J., M. Fu, and X. Feng. 2015. "Practical tactics for verifying C programs in Coq". In: *Proceedings of the 2015 Conference on Certified Programs and Proofs.* ACM. 97–108. DOI: 10.1145/2676724.2693162.

Cao, Q., L. Beringer, S. Gruetter, J. Dodds, and A. W. Appel. 2018. "VST-Floyd: A Separation Logic Tool to Verify Correctness of C Programs". *Journal of Automated Reasoning.* 61(1): 367–422. DOI: 10.1007/s10817-018-9457-5.

Caplan, J. E. and M. T. Harandi. 1995. "A Logical Framework for Software Proof Reuse". *SIGSOFT Softw. Eng. Notes*. 20(SI): 106–113. DOI: 10.1145/223427.211821.

Capretta, V. 2005. "General Recursion via Coinductive Types". *Logical Methods in Computer Science*. Volume 1, Issue 2(July). DOI: 10.2168/LMCS-1(2:1)2005.

Capretta, V. and A. P. Felty. 2007. "Combining de Bruijn Indices and Higher-Order Abstract Syntax in Coq". In: *Types for Proofs and Programs*. Berlin, Heidelberg: Springer. 63–77. DOI: 10.1007/978-3-540-74464-1_5.

Cardelli, L., S. Martini, J. C. Mitchell, and A. Scedrov. 1994. "An Extension of System F with Subtyping". *Inf. Comput.* 109(1-2): 4–56. DOI: 10.1006/inco.1994.1013.

Casinghino, C., V. Sjöberg, and S. Weirich. 2014. "Combining proofs and programs in a dependently typed language". In: *The 41st Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL '14, San Diego, CA, USA, January 20-21, 2014*. 33–46. DOI: 10.1145/2535838.2535883.

Celik, A., K. Palmskog, and M. Gligoric. 2017. "iCoq: Regression Proof Selection for Large-Scale Verification Projects". In: *32nd IEEE/ACM International Conference on Automated Software Engineering. ASE 2017*. 171–182. DOI: 10.1109/ASE.2017.8115630.

Chaieb, A. and T. Nipkow. 2008. "Proof synthesis and reflection for linear arithmetic". *Journal of Automated Reasoning*. 41(1): 33.

Chajed, T., J. Tassarotti, M. F. Kaashoek, and N. Zeldovich. 2019. "Argosy: Verifying Layered Storage Systems with Recovery Refinement". In: *Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation. PLDI 2019*. Phoenix, AZ, USA: ACM. 1054–1068. DOI: 10.1145/3314221.3314585.

Chan, M., J. Lehmann, and A. Bundy. 2011. "GALILEO: A system for automating ontology evolution". *ARCOE-11*: 46.

Charguéraud, A. 2010. "Program Verification Through Characteristic Formulae". In: *Proceedings of the 15th ACM SIGPLAN International Conference on Functional Programming. ICFP '10*. Baltimore, MD, USA: ACM. 321–332. DOI: 10.1145/1863543.1863590.

Charguéraud, A. 2011. "The Locally Nameless Representation". *Journal of Automated Reasoning*: 1–46. DOI: 10.1007/s10817-011-9225-2.

Chen, H., T. Chajed, A. Konradi, S. Wang, A. İleri, A. Chlipala, M. F. Kaashoek, and N. Zeldovich. 2017. "Verifying a High-performance Crash-safe File System Using a Tree Specification". In: *Proceedings of the Symposium on Operating Systems Principles*. 270–286. DOI: 10.1145/3132747.3132776.

Chen, H., D. Ziegler, T. Chajed, A. Chlipala, M. F. Kaashoek, and N. Zeldovich. 2015. "Using Crash Hoare Logic for Certifying the FSCQ File System". In: *Proceedings of the 25th Symposium on Operating Systems Principles. SOSP '15*. Monterey, California: ACM. 18–37. DOI: 10.1145/2815400.2815402.

Chlipala, A. 2008. "Parametric Higher-order Abstract Syntax for Mechanized Semantics". In: *Proceedings of the 13th ACM SIGPLAN International Conference on Functional Programming. ICFP '08*. Victoria, BC, Canada: ACM. 143–156. DOI: 10.1145/1411204.1411226.

Chlipala, A. 2011. "Mostly-automated verification of low-level programs in computational separation logic". In: *PLDI*. ACM. 234–245. DOI: 10.1145/1993498.1993526.

Chlipala, A. 2013a. *Certified Programming with Dependent Types: A Pragmatic Introduction to the Coq Proof Assistant*. Cambridge, MA, USA: MIT Press.

Chlipala, A. 2013b. "The Bedrock Structured Programming System: Combining Generative Metaprogramming and Hoare Logic in an Extensible Program Verifier". In: *Proceedings of the 18th ACM SIGPLAN International Conference on Functional Programming. ICFP '13*. Boston, MA, USA: ACM. 391–402. DOI: 10.1145/2500365.2500592.

Chlipala, A. 2017. "Formal Reasoning About Programs". URL: http://adam.chlipala.net/frap/.

Chlipala, A. 2018. "The Surprising Security Benefits of End-to-End Formal Proofs". URL: https://www.cccblog.org/2018/06/13/the-surprising-security-benefits-of-end-to-end-formal-proofs/.

Chlipala, A., B. Delaware, S. Duchovni, J. Gross, C. Pit-Claudel, S. Suriyakarn, P. Wang, and K. Ye. 2017. “The End of History: Using a Proof Assistant to Replace Language Design with Library Design”. In: *2nd Summit on Advances in Programming Languages (SNAPL 2017)*. Vol. 71. *Leibniz International Proceedings in Informatics (LIPIcs)*. Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. 3:1–3:15. DOI: 10.4230/LIPIcs.SNAPL.2017.3.

Chlipala, A., J. G. Malecha, G. Morrisett, A. Shinnar, and R. Wisnesky. 2009. “Effective interactive proofs for higher-order imperative programs”. In: *Proceeding of the 14th ACM SIGPLAN international conference on Functional programming (ICFP 2009)*. ACM. 79–90. DOI: 10.1145/1596550.1596565.

Christiansen, D. and E. Brady. 2016. “Elaborator Reflection: Extending Idris in Idris”. *SIGPLAN Not.* 51(9): 284–297. DOI: 10.1145/3022670.2951932.

Chrząszcz, J. 2003. “Implementing Modules in the Coq System”. In: *Theorem Proving in Higher Order Logics*. Berlin, Heidelberg: Springer. 270–286. DOI: 10.1007/10930755_18.

Church, A. 1936. “An Unsolvable Problem of Elementary Number Theory”. *American Journal of Mathematics*. 58(2): 345–363. DOI: 10.2307/2371054.

Church, A. 1940. “A formulation of the simple theory of types”. *Journal of Symbolic Logic*. 5(2): 56–68. DOI: 10.2307/2266170.

Church, A. 1941. *The calculi of lambda-conversion*. Princeton University Press.

Ciaffaglione, A. and I. Scagnetto. 2012. “A weak HOAS approach to the POPLMark Challenge”. In: *Proceedings Seventh Workshop on Logical and Semantic Frameworks, with Applications, LSFA 2012, Rio de Janeiro, Brazil, September 29-30, 2012*. 109–124. DOI: 10.4204/EPTCS.113.11.

Claret, G., L. d. C. G. Huesca, Y. Régis-Gianas, and B. Ziliani. 2013. “Lightweight proof by reflection using a posteriori simulation of effectful computation”. In: *International Conference on Interactive Theorem Proving*. Springer. 67–83. DOI: 10.1007/978-3-642-39634-2_8.

cody. 2015. “Why does Coq have Prop?” Theoretical Computer Science Stack Exchange. URL: http://cstheory.stackexchange.com/questions/21836/why-does-coq-have-prop.

Cohen, C. 2013. “Pragmatic Quotient Types in Coq”. In: *Interactive Theorem Proving*. Berlin, Heidelberg: Springer. 213–228. DOI: 10.1007/978-3-642-39634-2_17.

Cohen, C., T. Coquand, S. Huber, and A. Mörtberg. 2018. “Cubical Type Theory: A Constructive Interpretation of the Univalence Axiom”. In: *21st International Conference on Types for Proofs and Programs (TYPES 2015)*. Vol. 69. Schloss Dagstuhl. 5:1–5:34. DOI: 10.4230/LIPIcs.TYPES.2015.5.

Cohen, C., M. Dénès, and A. Mörtberg. 2013. “Refinements for Free!” In: *Certified Programs and Proofs*. Cham: Springer International Publishing. 147–162. DOI: 10.1007/978-3-319-03545-1_10.

Cohn, A. 1983. “The Equivalence of Two Semantic Definitions: A Case Study in LCF”. *SIAM Journal on Computing*. 12(2): 267–285. DOI: 10.1137/0212016.

CompCert Development Team. 2010. “Merge of the newmem and newextcalls branches”. URL: http://github.com/AbsInt/CompCert/commit/a74f6b45d72834b5b8417297017bd81424123d98.

Constable, R. L., 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. 1986. *Implementing Mathematics with the Nuprl Proof Development System*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc.

Cooper, K. and L. Torczon. 2011. *Engineering a compiler*. Elsevier. DOI: 10.1016/C2009-0-27982-7.

Coq Development Team. 2003. “interface GTK2 experimentale”. URL: http://github.com/coq/coq/commit/3fe1d8e4287.

Coq Development Team. 2017. “CoqStyle”. URL: https://github.com/coq/coq/wiki/CoqStyle.

Coq Development Team. 2017-2019. “CoqInTheClassroom”. URL: http://github.com/coq/coq/wiki/CoqInTheClassroom.

Coq Development Team. 2018-2019. “Preliminary compilation of critical bugs in stable releases of Coq”. URL: http://github.com/coq/coq/blob/master/dev/doc/critical-bugs.

Coq Development Team. 1999-2018a. "Coq Integrated Development Environment". URL: http://coq.inria.fr/refman/practical-tools/coqide.html.

Coq Development Team. 1999-2018b. "Tactics". URL: http://coq.inria.fr/refman/proof-engine/tactics.html.

Coq Development Team. 1999-2018c. "The Coq Commands". URL: http://coq.inria.fr/refman/practical-tools/coq-commands.html.

Coq Development Team. 1989-2019. "The Coq Proof Assistant". URL: http://coq.inria.fr.

Coq development team. 2018. "Coq OPAM Package Index". URL: https://coq.inria.fr/opam/www/.

CoqHoTT Development Team. 2015-2019. "The CoqHoTT Project". URL: http://coqhott.gforge.inria.fr/.

Coquand, T. 1994. "Infinite objects in type theory". In: *Types for Proofs and Programs*. Berlin, Heidelberg: Springer. 62–78. DOI: 10.1007/3-540-58085-9_72.

Coquand, T. and G. Huet. 1985. "Constructions: A higher order proof system for mechanizing mathematics". In: *EUROCAL '85*. Berlin, Heidelberg: Springer. 151–184. DOI: 10.1007/3-540-15983-5_13.

Coquand, T. and G. Huet. 1988. "The calculus of constructions". *Information and Computation*. 76(2): 95–120. DOI: 10.1016/0890-5401(88)90005-3.

Coquand, T. and C. Paulin-Mohring. 1990. "Inductively defined types". In: *COLOG-88*. Berlin, Heidelberg: Springer. 50–66. DOI: 10.1007/3-540-52335-9_47.

Corbineau, P. 2008. "A Declarative Language for the Coq Proof Assistant". In: *Types for Proofs and Programs: International Conference, TYPES 2007, Cividale des Friuli, Italy, May 2-5, 2007 Revised Selected Papers*. Berlin, Heidelberg: Springer. 69–84. DOI: 10.1007/978-3-540-68103-8_5.

Cornes, C., J. Courant, J.-C. Filliâtre, G. Huet, P. Manoury, C. Munoz, C. Murthy, C. Parent, C. Paulin-Mohring, A. Saibi, *et al.* 1995. "The Coq Proof assistant, Reference Manual, Version 5.10".

Cornes, C. and D. Terrasse. 1995. "Automating inversion of inductive predicates in Coq". In: *International Workshop on Types for Proofs and Programs*. Springer. 85–104. DOI: 10.1007/3-540-61780-9_64.

Crary, K. and R. Harper. 2014. "The Mechanization of Standard ML". URL: http://github.com/SMLFamily/The-Mechanization-of-Standard-ML.

Crégut, P. 1999-2018. "Omega: A solver for quantifier-free problems in Presburger Arithmetic". URL: http://coq.inria.fr/refman/addendum/omega.html.

Cruz-Filipe, L. and P. Letouzey. 2006. "A Large-Scale Experiment in Executing Extracted Programs". *Electronic Notes in Theoretical Computer Science*. 151(1): 75–91. DOI: 10.1016/j.entcs.2005.11.024.

Cruz-Filipe, L. and B. Spitters. 2003. "Program Extraction from Large Proof Developments". In: *Theorem Proving in Higher Order Logics: 16th International Conference, TPHOLs 2003, Rome, Italy, September 8-12, 2003. Proceedings*. Berlin, Heidelberg: Springer. 205–220. DOI: 10.1007/10930755_14.

Curien, R. 1995. "Tools for proof by analogy". *Theses*. Université Henri Poincaré - Nancy 1. URL: https://hal.univ-lorraine.fr/tel-01748604.

Curry, H. B. 1934. "Functionality in Combinatory Logic". *Proceedings of the National Academy of Sciences of the United States of America*. 20(11): 584–590. URL: http://www.jstor.org/stable/86796.

Czajka, Ł. and C. Kaliszyk. 2018. "Hammer for Coq: Automation for Dependent Type Theory". *Journal of Automated Reasoning*. 61(1): 423–453. DOI: 10.1007/s10817-018-9458-4.

Czajka, Ł., C. Kaliszyk, and B. Ekici. 2018. "CoqHammer: Automation for Dependent Type Theory". URL: http://cl-informatik.uibk.ac.at/cek/coqhammer/.

Dagand, P. 2017. "The essence of ornaments". *J. Funct. Program.* 27: e9. DOI: 10.1017/S0956796816000356.

Dahn, B. I., J. Gehne, T. Honigmann, and A. Wolf. 1997. "Integration of automated and interactive theorem proving in ILF". In: *International Conference on Automated Deduction*. Springer. 57–60. DOI: 10.1007/3-540-63104-6_7.

Dam, M., R. Guanciale, N. Khakpour, H. Nemati, and O. Schwarz. 2013. "Formal Verification of Information Flow Security for a Simple ARM-based Separation Kernel". In: *Proceedings of the 2013 ACM SIGSAC Conference on Computer and Communications Security. CCS '13*. Berlin, Germany: ACM. 223–234. DOI: 10.1145/2508859.2516702.

Danielsson, N. A. 2012. “Operational Semantics Using the Partiality Monad”. In: *Proceedings of the 17th ACM SIGPLAN International Conference on Functional Programming. ICFP ’12.* Copenhagen, Denmark: ACM. 127–138. DOI: 10.1145/2364527.2364546.

Darmochwał, A. 1990. “Finite sets”. *Formalized Mathematics.* 1(1): 165–167.

Davis, J. and M. O. Myreen. 2015. “The Reflective Milawa Theorem Prover is Sound (Down to the Machine Code that Runs it)”. *Journal of Automated Reasoning.* 55(2): 117–183. DOI: 10.1007/s10817-015-9324-6.

de Bruijn, N. G. 1970. “The mathematical language Automath, its usage, and some of its extensions”. In: *Symposium on Automatic Demonstration.* Berlin, Heidelberg: Springer. 29–61. DOI: 10.1007/BFb0060623.

de Bruijn, N. G. 1972. “Lambda calculus notation with nameless dummies, a tool for automatic formula manipulation, with application to the Church-Rosser theorem”. *Indagationes Mathematicae (Proceedings).* 75(5): 381–392. DOI: 10.1016/1385-7258(72)90034-0.

de Bruijn, N. G. 1994. “A Survey of the Project Automath”. In: *Selected Papers on Automath.* Ed. by R. Nederpelt, J. Geuvers, and R. de Vrijer. Vol. 133. *Studies in Logic and the Foundations of Mathematics.* Elsevier. 141–161. DOI: 10.1016/S0049-237X(08)70203-9.

de Moura, L., S. Kong, J. Avigad, F. van Doorn, and J. von Raumer. 2015. “The Lean Theorem Prover (System Description)”. In: *Automated Deduction - CADE-25: 25th International Conference on Automated Deduction, Berlin, Germany, August 1-7, 2015, Proceedings.* Cham: Springer International Publishing. 378–388. DOI: 10.1007/978-3-319-21401-6_26.

De Roever, W.-P., K. Engelhardt, and K.-H. Buth. 1998. *Data refinement: model-oriented proof methods and their comparison.* Vol. 47. Cambridge University Press.

DeepSpec Team. 2013-2019. “DeepSpec Project”. URL: https://deepspec.org.

Delahaye, D. 2000. "A Tactic Language for the System Coq". In: *Logic for Programming and Automated Reasoning: 7th International Conference, LPAR 2000 Reunion Island, France, November 6–10, 2000 Proceedings*. Berlin, Heidelberg: Springer. 85–95. DOI: 10.1007/3-540-44404-1_7.

Delaware, B., W. Cook, and D. Batory. 2011. "Product Lines of Theorems". In: *Proceedings of the 2011 ACM International Conference on Object Oriented Programming Systems Languages and Applications. OOPSLA '11*. Portland, OR, USA: ACM. 595–608. DOI: 10.1145/2048066.2048113.

Delaware, B., S. Keuchel, T. Schrijvers, and B. C. Oliveira. 2013a. "Modular Monadic Meta-theory". In: *Proceedings of the 18th ACM SIGPLAN International Conference on Functional Programming. ICFP '13*. Boston, MA, USA: ACM. 319–330. DOI: 10.1145/2500365.2500587.

Delaware, B., C. Pit-Claudel, J. Gross, and A. Chlipala. 2015. "Fiat: Deductive Synthesis of Abstract Data Types in a Proof Assistant". In: *Proceedings of the 42Nd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages. POPL '15*. Mumbai, India: ACM. 689–700. DOI: 10.1145/2676726.2677006.

Delaware, B., B. C. d. S. Oliveira, and T. Schrijvers. 2013b. "Meta-theory à La Carte". In: *Proceedings of the 40th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages. POPL '13*. Rome, Italy: ACM. 207–218. DOI: 10.1145/2429069.2429094.

Demers, F.-N. and J. Malenfant. 1995. "Reflection in logic, functional and object-oriented programming: A Short Comparative Study". In: *In IJCAI '95 Workshop on Reflection and Metalevel Architectures and their Applications in AI*. 29–38.

DeMillo, R. A., R. J. Lipton, and A. J. Perlis. 1977. "Social Processes and Proofs of Theorems and Programs". In: *Proceedings of the 4th ACM SIGACT-SIGPLAN Symposium on Principles of Programming Languages. POPL '77*. Los Angeles, California: ACM. 206–214. DOI: 10.1145/512950.512970.

Devriese, D. and F. Piessens. 2011. “On the Bright Side of Type Classes: Instance Arguments in Agda”. In: *Proceedings of the 16th ACM SIGPLAN International Conference on Functional Programming. ICFP ’11.* Tokyo, Japan: ACM. 143–155. DOI: 10.1145/2034773.2034796.

Diehl, L., D. Firsov, and A. Stump. 2018. “Generic Zero-cost Reuse for Dependent Types”. *Proc. ACM Program. Lang.* 2(ICFP): 104:1–104:30. DOI: 10.1145/3236799.

Dietl, W., S. Dietzel, M. D. Ernst, N. Mote, B. Walker, S. Cooper, T. Pavlik, and Z. Popović. 2012. “Verification games: Making verification fun”. In: *Proceedings of the 14th Workshop on Formal Techniques for Java-like Programs.* ACM. 42–49. DOI: 10.1145/2318202.2318210.

Dietrich, D. 2011. *Assertion level proof planning with compiled strategies.*

Dietrich, D., I. Whiteside, and D. Aspinall. 2013. “Polar: A Framework for Proof Refactoring”. In: *Logic for Programming, Artificial Intelligence, and Reasoning.* Berlin, Heidelberg: Springer. 776–791. DOI: 10.1007/978-3-642-45221-5_52.

Dijkstra, E. W. 1975. “Guarded Commands, Nondeterminacy and Formal Derivation of Programs”. *Commun. ACM.* 18(8): 453–457. DOI: 10.1145/360933.360975.

Dixon, L. and J. Fleuriot. 2003. “IsaPlanner: A Prototype Proof Planner in Isabelle”. In: *Automated Deduction – CADE-19.* Berlin, Heidelberg: Springer. 279–283. DOI: 10.1007/978-3-540-45085-6_22.

Dockins, R. and A. Hobor. 2010. “A theory of termination via indirection”. In: *Proceedings of the Dagstuhl Seminar on Modelling, Controlling and Reasoning about State.* Vol. 10351. Dagstuhl, Germany. 166–177.

Doczkal, C. and G. Smolka. 2018. “Regular Language Representations in the Constructive Type Theory of Coq”. *Journal of Automated Reasoning.* 61(1): 521–553. DOI: 10.1007/s10817-018-9460-x.

Dramnesc, I., T. Jebelean, and S. Stratulat. 2015. “Theory exploration of binary trees”. In: *Intelligent Systems and Informatics (SISY), 2015 IEEE 13th International Symposium on.* IEEE. 139–144. DOI: 10.1109/SISY.2015.7325367.

Dreyer, D., R. Harper, M. M. T. Chakravarty, and G. Keller. 2007. "Modular Type Classes". In: *Proceedings of the 34th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages. POPL '07.* Nice, France: ACM. 63–70. DOI: 10.1145/1190216.1190229.

Ebner, G., S. Ullrich, J. Roesch, J. Avigad, and L. de Moura. 2017. "A Metaprogramming Framework for Formal Verification". *Proc. ACM Program. Lang.* 1(ICFP): 34:1–34:29. DOI: 10.1145/3110278.

Eclipse Foundation. 2001-2019. "Eclipse". URL: http://www.eclipse.org/ide/.

Elphinstone, K. and G. Heiser. 2013. "From L3 to seL4 What Have We Learnt in 20 Years of L4 Microkernels?" In: *Proceedings of the Twenty-Fourth ACM Symposium on Operating Systems Principles. SOSP '13.* Farminton, PA: ACM. 133–150. DOI: 10.1145/2517349.2522720.

Erbsen, A., J. Philipoom, J. Gross, R. Sloan, and A. Chlipala. 2019. "Simple High-Level Code For Cryptographic Arithmetic – With Proofs, Without Compromises". In: *IEEE Symposium on Security and Privacy.* DOI: 10.1109/SP.2019.00005.

Escardó, M. H. 2018. "A self-contained, brief and complete formulation of Voevodsky's Univalence Axiom". *CoRR.* abs/1803.02294.

Esparza, J., P. Lammich, R. Neumann, T. Nipkow, A. Schimpf, and J.-G. Smaus. 2013. "A fully verified executable LTL model checker". In: *International Conference on Computer Aided Verification.* Springer. 463–478.

Faithfull, A., J. Bengtson, E. Tassi, and C. Tankink. 2018. "Coqoon: An IDE for interactive proof development in Coq". *International Journal on Software Tools for Technology Transfer.* 20(2): 125–137. DOI: 10.1007/s10009-017-0457-2.

Fallenstein, B. and R. Kumar. 2015. "Proof-producing reflection for HOL". In: *International Conference on Interactive Theorem Proving.* Springer. 170–186. DOI: 10.1007/978-3-319-22102-1_11.

Feferman, S. 2005. "Predicativity". In: *Oxford Handbook of Philosophy of Mathematics and Logic.* Ed. by S. Shapiro. Oxford University Press. 590–624.

Felty, A. P., A. Momigliano, and B. Pientka. 2015. "The Next 700 Challenge Problems for Reasoning with Higher-Order Abstract Syntax Representations: Part 1-A Common Infrastructure for Benchmarks". *CoRR*. abs/1503.06095.

Felty, A. and D. Howe. 1994. "Generalization and reuse of tactic proofs". In: *Logic Programming and Automated Reasoning: 5th International Conference. LPAR '94*. Berlin, Heidelberg: Springer. 1–15. DOI: 10.1007/3-540-58216-9_25.

Felty, A. and A. Momigliano. 2012. "Hybrid: A Definitional Two-Level Approach to Reasoning with Higher-Order Abstract Syntax". *Journal of Automated Reasoning*. 48(1): 43–105. DOI: 10.1007/s10817-010-9194-x.

Felty, A., A. Momigliano, and B. Pientka. 2018. "Benchmarks for reasoning with syntax trees containing binders and contexts of assumptions". *Mathematical Structures in Computer Science*. 28(9): 1507–1540. DOI: 10.1017/S0960129517000093.

Felty, A. and B. Pientka. 2010. "Reasoning with Higher-Order Abstract Syntax and Contexts: A Comparison". In: *Interactive Theorem Proving*. Berlin, Heidelberg: Springer. 227–242. DOI: 10.1007/978-3-642-14052-5_17.

Feng, X. 2009. "Local rely-guarantee reasoning". In: *POPL*. ACM. 315–327. DOI: 10.1145/1480881.1480922.

Feng, X. and Z. Shao. 2005. "Modular verification of concurrent assembly code with dynamic thread creation and termination". In: *Proceedings of the 10th ACM SIGPLAN International Conference on Functional Programming (ICFP 2005)*. ACM. 254–267. DOI: 10.1145/1086365.1086399.

Filliâtre, J.-C. and P. Letouzey. 2004. "Functors for Proofs and Programs". In: *Programming Languages and Systems*. Berlin, Heidelberg: Springer. 370–384. DOI: 10.1007/978-3-540-24725-8_26.

Filman, R., T. Elrad, S. Clarke, and M. Akşit. 2004. *Aspect-oriented Software Development*. First. Addison-Wesley Professional.

Floyd, R. W. 1967. "Assigning Meanings to Programs". *Proceedings of Symposium on Applied Mathematics*. 19.

Fox, A. and M. O. Myreen. 2010. "A Trustworthy Monadic Formalization of the ARMv7 Instruction Set Architecture". In: *Interactive Theorem Proving*. Berlin, Heidelberg: Springer. 243–258. DOI: 10.1007/978-3-642-14052-5_18.

França, R. B., D. Favre-Felix, X. Leroy, M. Pantel, and J. Souyris. 2011. "Towards Formally Verified Optimizing Compilation in Flight Control Software". In: *Bringing Theory to Practice: Predictability and Performance in Embedded Systems*. Vol. 18. *OpenAccess Series in Informatics (OASIcs)*. Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. 59–68. DOI: 10.4230/OASIcs.PPES.2011.59.

Frege, G. 1893. *Grundgesetze der Arithmetik*. Jena: Verlag Hermann Pohle.

Gallego Arias, E. J., B. Pin, and P. Jouvelot. 2017. "jsCoq: Towards Hybrid Theorem Proving Interfaces". In: *Proceedings of the 12th Workshop on User Interfaces for Theorem Provers, Coimbra, Portugal, 2nd July 2016*. Vol. 239. *Electronic Proceedings in Theoretical Computer Science*. Open Publishing Association. 15–27. DOI: 10.4204/EPTCS.239.2.

Garillot, F. 2011. "Generic Proof Tools and Finite Group Theory". *PhD thesis*. Palaiseau, France: École Polytechnique.

Garillot, F., G. Gonthier, A. Mahboubi, and L. Rideau. 2009. "Packaging Mathematical Structures". In: *TPHOL*. Vol. 5674. *LNCS*. Springer. 327–342. DOI: 10.1007/978-3-642-03359-9_23.

Gauthier, T. and C. Kaliszyk. 2014. "Matching concepts across HOL libraries". In: *Intelligent Computer Mathematics*. Springer. 267–281. DOI: 10.1007/978-3-319-08434-3_20.

Gauthier, T. and C. Kaliszyk. 2015. "Sharing HOL4 and HOL Light Proof Knowledge". In: *Logic for Programming, Artificial Intelligence, and Reasoning: 20th International Conference, LPAR-20 2015, Suva, Fiji, November 24-28, 2015, Proceedings*. Berlin, Heidelberg: Springer. 372–386. DOI: 10.1007/978-3-662-48899-7_26.

Gauthier, T. and C. Kaliszyk. 2019. "Aligning concepts across proof assistant libraries". *Journal of Symbolic Computation*. 90: 89–123. Symbolic Computation in Software Science. DOI: 10.1016/j.jsc.2018.04.005.

Gauthier, T., C. Kaliszyk, and J. Urban. 2017. "TacticToe: Learning to Reason with HOL4 Tactics". In: *LPAR-21. 21st International Conference on Logic for Programming, Artificial Intelligence and Reasoning*. Vol. 46. *EPiC Series in Computing*. EasyChair. 125–143. DOI: 10.29007/ntlb.

Gauthier, T., C. Kaliszyk, J. Urban, R. Kumar, and M. Norrish. 2018. "Learning to Prove with Tactics". *CoRR*. abs/1804.00596. arXiv: 1804.00596.

Geuvers, H. 2009. "Proof assistants: History, ideas and future". *Sadhana*. 34(1): 3–25. DOI: 10.1007/s12046-009-0001-5.

Geuvers, H., R. Pollack, F. Wiedijk, and J. Zwanenburg. 2002a. "A Constructive Algebraic Hierarchy in Coq". *Journal of Symbolic Computation*. 34(4): 271–286. DOI: 10.1006/jsco.2002.0552.

Geuvers, H., F. Wiedijk, and J. Zwanenburg. 2002b. "A Constructive Proof of the Fundamental Theorem of Algebra Without Using the Rationals". In: *Types for Proofs and Programs. TYPES '00*. Berlin, Heidelberg: Springer. 96–111. DOI: 10.1007/3-540-45842-5_7.

Giero, M., F. Wiedijk, M. Giero, and F. Wiedijk. 2003. "MMode, a Mizar mode for the proof assistant Coq". URL: http://www.cs.kun.nl/~freek/mmode/mmode.pdf.

Giménez, E. 1995. "Codifying guarded definitions with recursive schemes". In: *Types for Proofs and Programs*. Berlin, Heidelberg: Springer. 39–59. DOI: 10.1007/3-540-60579-7_3.

GitHub. 2014-2019. "Atom". URL: http://atom.io/.

GNU Project. 1985-2019. "GNU Emacs". URL: http://www.gnu.org/software/emacs/.

Gödel, K. 1930. "Die Vollständigkeit der Axiome des logischen Funktionenkalküls". *Monatshefte für Mathematik und Physik*. 37(1): 349–360. DOI: 10.1007/BF01696781.

Gödel, K. 1931. "Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme I". *Monatshefte für Mathematik und Physik*. 38(1): 173–198. DOI: 10.1007/BF01700692.

Gomes, V. B., M. Kleppmann, D. P. Mulligan, and A. R. Beresford. 2017. "Verifying strong eventual consistency in distributed systems". *Proceedings of the ACM on Programming Languages*. 1(OOPSLA): 109. DOI: 10.1145/3133933.

Gonthier, G. 2008. “Formal proof—the four-color theorem”. *Notices of the American Mathematical Society*. 55(11): 1382–1393. URL: http://www.ams.org/notices/200811/tx081101382p.pdf.

Gonthier, G., A. Asperti, J. Avigad, Y. Bertot, C. Cohen, F. Garillot, S. Le Roux, A. Mahboubi, R. O’Connor, S. Ould Biha, I. Pasca, L. Rideau, A. Solovyev, E. Tassi, and L. Théry. 2013. “A Machine-Checked Proof of the Odd Order Theorem”. In: *Interactive Theorem Proving: 4th International Conference, ITP 2013, Rennes, France, July 22-26, 2013. Proceedings*. Berlin, Heidelberg: Springer. 163–179. DOI: 10.1007/978-3-642-39634-2_14.

Gonthier, G. and A. Mahboubi. 2010. “An introduction to small scale reflection in Coq”. *Journal of Formalized Reasoning*. 3(2): 95–152. DOI: 10.6092/issn.1972-5787/1979.

Gonthier, G. and E. Tassi. 2012. “A Language of Patterns for Subterm Selection”. In: *Interactive Theorem Proving*. Berlin, Heidelberg: Springer. 361–376. DOI: 10.1007/978-3-642-32347-8_25.

Gonthier, G., B. Ziliani, A. Nanevski, and D. Dreyer. 2011. “How to Make Ad Hoc Proof Automation Less Ad Hoc”. In: *Proceedings of the 16th ACM SIGPLAN International Conference on Functional Programming. ICFP ’11*. Tokyo, Japan: ACM. 163–175. DOI: 10.1145/2034773.2034798.

Gordon, A. D. 1994. “A mechanisation of name-carrying syntax up to alpha-conversion”. In: *Higher Order Logic Theorem Proving and Its Applications*. Berlin, Heidelberg: Springer. 413–425. DOI: 10.1007/3-540-57826-9_152.

Gordon, C. S., M. D. Ernst, D. Grossman, and M. J. Parkinson. 2017. “Verifying Invariants of Lock-Free Data Structures with Rely-Guarantee and Refinement Types”. *ACM Trans. Program. Lang. Syst.* 39(3): 11:1–11:54. DOI: 10.1145/3064850.

Gordon, M. J. C. 2000. “Proof, Language, and Interaction”. In: ed. by G. Plotkin, C. Stirling, and M. Tofte. Cambridge, MA, USA: MIT Press. Chap. From LCF to HOL: A Short History. 169–185.

Gordon, M. J. C. and T. F. Melham, eds. 1993. *Introduction to HOL: A Theorem Proving Environment for Higher Order Logic*. New York, NY, USA: Cambridge University Press.

Gordon, M. J. C., R. Milner, L. Morris, M. C. Newey, and C. P. Wadsworth. 1978. "A Metalanguage for Interactive Proof in LCF". In: *Proceedings of the 5th ACM SIGACT-SIGPLAN Symposium on Principles of Programming Languages. POPL '78.* Tucson, Arizona: ACM. 119–130. DOI: 10.1145/512760.512773.

Gordon, M. J. C., R. Milner, and C. P. Wadsworth. 1979. "Edinburgh LCF: A Mechanised Logic of Computation". In: *Lecture Notes in Computer Science.* Vol. 78. DOI: 10.1007/3-540-09724-4.

Graham, P. 1996. *ANSI Common Lisp.* Prentice Hall Press.

Grégoire, B. and A. Mahboubi. 2005. "Proving equalities in a commutative ring done right in Coq". In: *International Conference on Theorem Proving in Higher Order Logics.* Springer. 98–113. DOI: 10.1007/11541868_7.

Gu, R., J. Koenig, T. Ramananandro, Z. Shao, X. Wu, S. Weng, H. Zhang, and Y. Guo. 2015. "Deep Specifications and Certified Abstraction Layers". In: *POPL.* ACM. 595–608. DOI: 10.1145/2676726.2676975.

Gu, R., Z. Shao, H. Chen, X. (Wu, J. Kim, V. Sjöberg, and D. Costanzo. 2016. "CertiKOS: An Extensible Architecture for Building Certified Concurrent OS Kernels". In: *OSDI.* USENIX Association. 653–669.

Gu, R., Z. Shao, J. Kim, X. Wu, J. Koenig, V. Sjöberg, H. Chen, D. Costanzo, and T. Ramananandro. 2018. "Certified concurrent abstraction layers". In: *PLDI.* ACM. 646–661. DOI: 10.1145/3192366.3192381.

Guanciale, R., H. Nemati, M. Dam, and C. Baumann. 2016. "Provably secure memory isolation for Linux on ARM". *Journal of Computer Security.* 24(6): 793–837. DOI: 10.3233/JCS-160558.

Haftmann, F., A. Krauss, O. Kunčar, and T. Nipkow. 2013. "Data Refinement in Isabelle/HOL". In: *Interactive Theorem Proving.* Berlin, Heidelberg: Springer. 100–115. DOI: 10.1007/978-3-642-39634-2_10.

Haftmann, F. and T. Nipkow. 2010. "Code Generation via Higher-order Rewrite Systems". In: *Proceedings of the 10th International Conference on Functional and Logic Programming. FLOPS'10.* Sendai, Japan: Springer-Verlag. 103–117. DOI: 10.1007/978-3-642-12251-4_9.

Haftmann, F. and M. Wenzel. 2007. “Constructive Type Classes in Isabelle”. In: *Types for Proofs and Programs.* Ed. by T. Altenkirch and C. McBride. Berlin, Heidelberg: Springer. 160–174. DOI: 10.1007/978-3-540-74464-1_11.

Haftmann, F. and M. Wenzel. 2009. “Local Theory Specifications in Isabelle/Isar”. In: *Types for Proofs and Programs.* Berlin, Heidelberg: Springer. 153–168. DOI: 10.1007/978-3-642-02444-3_10.

Hales, T. C. 2007. “The Jordan curve theorem, formally and informally”. *American Mathematical Monthly.* 114(10): 882–894.

Hales, T. C., J. Harrison, S. McLaughlin, T. Nipkow, S. Obua, and R. Zumkeller. 2011. “A Revision of the Proof of the Kepler Conjecture”. In: *The Kepler Conjecture: The Hales-Ferguson Proof.* New York, NY: Springer New York. 341–376. DOI: 10.1007/978-1-4614-1129-1_9.

Hales, T., M. Adams, G. Bauer, T. D. Dang, J. Harrison, L. T. Hoang, C. Kaliszyk, V. Magron, S. McLaughlin, T. T. Nguyen, Q. T. Nguyen, T. Nipkow, S. Obua, J. Pleso, J. Rute, A. Solovyev, T. H. A. Ta, N. T. Tran, T. D. Trieu, J. Urban, K. Vu, and R. Zumkeller. 2017. “A Formal Proof of the Kepler Conjecture”. *Forum of Mathematics, Pi.* 5. DOI: 10.1017/fmp.2017.1.

Harper, R. 1999. “Proof-directed Debugging”. *J. Funct. Program.* 9(4): 463–469. DOI: 10.1017/S0956796899003378.

Harper, R. 2011. “Modules Matter Most”. URL: https://existentialtype.wordpress.com/2011/04/16/modules-matter-most/.

Harper, R. 2016. *Practical foundations for programming languages.* 2nd. Cambridge, UK: Cambridge University Press. DOI: 10.1017/CBO9781316576892.

Harper, R. and K. Crary. 2019. Personal communication.

Harper, R., F. Honsell, and G. Plotkin. 1993. “A Framework for Defining Logics”. *J. ACM.* 40(1): 143–184. DOI: 10.1145/138027.138060.

Harper, R., F. Honsell, and G. D. Plotkin. 1987. “A Framework for Defining Logics”. In: *Proceedings of the Second Annual IEEE Symposium on Logic in Computer Science (LICS 1987).* Ithaca, NY, USA: IEEE Computer Society Press. 194–204.

Harper, R. and D. R. Licata. 2007. “Mechanizing metatheory in a logical framework”. *Journal of Functional Programming*. 17(4-5): 613–673. DOI: 10.1017/S0956796807006430.

Harrison, J. 1995. “Metatheory and Reflection in Theorem Proving: A Survey and Critique”. *Technical Report* No. CRC-053. Millers Yard, Cambridge, UK: SRI Cambridge. URL: http://www.cl.cam.ac.uk/~jrh13/papers/reflect.dvi.gz.

Harrison, J. 1996. “A Mizar Mode for HOL”. In: *Theorem Proving in Higher Order Logics*. Vol. 1125. Turku, Finland: Springer. 203–220. DOI: 10.1007/BFb0105406.

Harrison, J., J. Urban, and F. Wiedijk. 2014. “History of Interactive Theorem Proving”. In: *Computational Logic*. Ed. by J. H. Siekmann. Vol. 9. *Handbook of the History of Logic*. No. Supplement C. North-Holland. 135–214. DOI: 10.1016/B978-0-444-51624-4.50004-6.

Hasker, R. W. and U. S. Reddy. 1992. “Generalization at higher types”. In: *Proceedings of the Workshop on the λProlog Programming Language*. 257–271.

Hemer, D., I. Hayes, and P. Strooper. 2001. “Refinement Calculus for Logic Programming in Isabelle/HOL”. In: *Theorem Proving in Higher Order Logics*. Berlin, Heidelberg: Springer. 249–260. DOI: 10.1007/3-540-44755-5_18.

Heras, J. and E. Komendantskaya. 2013. “ML4PG in Computer Algebra Verification”. In: *Intelligent Computer Mathematics*. Berlin, Heidelberg: Springer. 354–358. DOI: 10.1007/978-3-642-39320-4_28.

Heras, J. and E. Komendantskaya. 2014. “Recycling Proof Patterns in Coq: Case Studies”. *Mathematics in Computer Science*. 8(1): 99–116. DOI: 10.1007/s11786-014-0173-1.

Heras, J., E. Komendantskaya, M. Johansson, and E. Maclean. 2013. “Proof-Pattern Recognition and Lemma Discovery in ACL2”. In: *Logic for Programming, Artificial Intelligence, and Reasoning: 19th International Conference, LPAR-19, Stellenbosch, South Africa, December 14-19, 2013. Proceedings*. Berlin, Heidelberg: Springer. 389–406. DOI: 10.1007/978-3-642-45221-5_27.

Heyting, A. 1956. “Intuitionism. An introduction”.

Hoare, C. A. R. 1969. “An Axiomatic Basis for Computer Programming”. *Commun. ACM*. 12(10): 576–580. DOI: 10.1145/363235.363259.

Hoare, C. A. R. 1971. “Proof of a Program: FIND”. *Commun. ACM.* 14(1): 39–45. DOI: 10.1145/362452.362489.

HOL Development Team. 2018. “The HOL System TUTORIAL”. URL: http://sourceforge.net/projects/hol/files/hol/kananaskis-12/kananaskis-12-tutorial.pdf?download.

HOL Development Team. 2016-2018. “Running hol”. URL: https://hol-theorem-prover.org/guidebook/#running-hol.

HOL Light Development Team. 1996-2019. “HOL Light”. URL: http://www.cl.cam.ac.uk/~jrh13/hol-light.

Homeier, P. V. 2005. “A Design Structure for Higher Order Quotients”. In: *In Proc. of the 18th International Conference on Theorem Proving in Higher Order Logics (TPHOLs), volume 3603 of LNCS.* 130–146. DOI: 10.1007/11541868_9.

Howard, W. A. 1980. “The formulae-as-types notion of constructions”. In: *To H. B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism.* Ed. by J. P. Seldin and J. R. Hindley. Academic Press.

Howe, D. J. 1988. “Computational metatheory in Nuprl”. In: *International Conference on Automated Deduction.* Springer. 238–257. DOI: 10.1007/BFb0012835.

Huang, D., P. Dhariwal, D. Song, and I. Sutskever. 2019. “GamePad: A Learning Environment for Theorem Proving”. In: *International Conference on Learning Representations.* URL: https://openreview.net/forum?id=r1xwKoR9Y7.

Huffman, B. and O. Kunčar. 2013. “Lifting and Transfer: A Modular Design for Quotients in Isabelle/HOL”. In: *Certified Programs and Proofs: Third International Conference. CPP 2013.* Cham: Springer International Publishing. 131–146. DOI: 10.1007/978-3-319-03545-1_9.

Hupel, L. and T. Nipkow. 2018. “A Verified Compiler from Isabelle/HOL to CakeML”. In: *Programming Languages and Systems.* Cham: Springer International Publishing. 999–1026. DOI: 10.1007/978-3-319-89884-1_35.

Hurd, J. 1999. “Integrating Gandalf and HOL”. In: *International Conference on Theorem Proving in Higher Order Logics.* Springer. 311–321. DOI: 10.1007/3-540-48256-3_21.

Hurd, J. 2003. “First-order proof tactics in higher-order logic theorem provers”. *Design and Application of Strategies/Tactics in Higher Order Logics, number NASA/CP-2003-212448 in NASA Technical Reports*: 56–68.

Igarashi, A., B. C. Pierce, and P. Wadler. 2001. “Featherweight Java: A minimal core calculus for Java and GJ”. *ACM Trans. Program. Lang. Syst.* 23(3): 396–450. DOI: 10.1145/503502.503505.

Inria. 1999-2018. “The Tactic Language”. URL: https://coq.inria.fr/refman/proof-engine/ltac.html.

Ireland, A. 1996. “Productive Use of Failure in Inductive Proof”. *J. Autom. Reasoning*. 16(1-2): 79–111. DOI: 10.1007/BF00244460.

Irvine, A. D. 2016. “Principia Mathematica”. In: *The Stanford Encyclopedia of Philosophy*. Ed. by E. N. Zalta. Winter 2016. Metaphysics Research Lab, Stanford University. URL: https://plato.stanford.edu/archives/win2016/entries/principia-mathematica/.

Irvine, A. D. and H. Deutsch. 2016. “Russell’s Paradox”. In: *The Stanford Encyclopedia of Philosophy*. Ed. by E. N. Zalta. Winter 2016. Metaphysics Research Lab, Stanford University. URL: https://plato.stanford.edu/archives/win2016/entries/russell-paradox/.

Irving, G., C. Szegedy, A. A. Alemi, N. Een, F. Chollet, and J. Urban. 2016. “DeepMath - Deep Sequence Models for Premise Selection”. In: *Advances in Neural Information Processing Systems 29*. Curran Associates, Inc. 2235–2243. URL: http://papers.nips.cc/paper/6280-deepmath-deep-sequence-models-for-premise-selection.pdf.

Isabelle Development Team. 1994-2019. “Isabelle”. URL: http://isabelle.in.tum.de.

Jang, D., Z. Tatlock, and S. Lerner. 2012. “Establishing Browser Security Guarantees Through Formal Shim Verification”. In: *Proceedings of the 21st USENIX Conference on Security Symposium. Security’12*. Bellevue, WA: USENIX Association. 8–8.

Jeffery, R., M. Staples, J. Andronick, G. Klein, and T. Murray. 2015. “An empirical research agenda for understanding formal methods productivity”. *Information and Software Technology*. 60: 102–112. DOI: 10.1016/j.infsof.2014.11.005.

JetBrains. 2001-2019. “IntelliJ IDEA”. URL: http://www.jetbrains.com/idea/.

Johansson, M. 2017. “Automated Theory Exploration for Interactive Theorem Proving”. In: *International Conference on Interactive Theorem Proving*. Springer. 1–11. DOI: 10.1007/978-3-319-66107-0_1.

Johansson, M., D. Rosén, N. Smallbone, and K. Claessen. 2014. “Hipster: Integrating Theory Exploration in a Proof Assistant”. *CoRR*. abs/1405.3426.

Johnsen, E. B. and C. Lüth. 2004. “Theorem Reuse by Proof Term Transformation”. In: *Theorem Proving in Higher Order Logics: 17th International Conference, TPHOLs 2004, Park City, Utah, USA, September 14-17, 2004. Proceedings*. Berlin, Heidelberg: Springer. 152–167. DOI: 10.1007/978-3-540-30142-4_12.

Jones, C. B. 1983. “Specification and Design of (Parallel) Programs”. In: *IFIP Congress*. 321–332.

Jourdan, J.-H., F. Pottier, and X. Leroy. 2012. “Validating LR(1) Parsers”. In: *Programming Languages and Systems*. Berlin, Heidelberg: Springer. 397–416. DOI: 10.1007/978-3-642-28869-2_20.

Jung, R., J.-H. Jourdan, R. Krebbers, and D. Dreyer. 2017. “RustBelt: Securing the Foundations of the Rust Programming Language”. *Proc. ACM Program. Lang.* 2(POPL): 66:1–66:34. DOI: 10.1145/3158154.

Jung, R., R. Krebbers, J.-H. Jourdan, A. Bizjak, L. Birkedal, and D. Dreyer. 2018. “Iris from the ground up: A modular foundation for higher-order concurrent separation logic”. *Journal of Functional Programming*. 28: e20. DOI: 10.1017/S0956796818000151.

Jung, R., D. Swasey, F. Sieczkowski, K. Svendsen, A. Turon, L. Birkedal, and D. Dreyer. 2015. “Iris: Monoids and Invariants as an Orthogonal Basis for Concurrent Reasoning”. In: *POPL*. ACM. 637–650. DOI: 10.1145/2676726.2676980.

Kadoda, G. F., R. G. Stone, and D. Diaper. 1999. “Desirable features of educational theorem provers - a cognitive dimensions viewpoint”. In: *PPIG*. Psychology of Programming Interest Group. 4.

Kaiser, J.-O., B. Ziliani, R. Krebbers, Y. Régis-Gianas, and D. Dreyer. 2018. “Mtac2: Typed Tactics for Backward Reasoning in Coq”. *Proc. ACM Program. Lang.* 2(ICFP): 78:1–78:31. DOI: 10.1145/3236773.

Kaivola, R. and K. Kohatsu. 2003. "Proof engineering in the large: formal verification of Pentium 4 floating-point divider". *International Journal on Software Tools for Technology Transfer*. 4(3): 323–334. DOI: 10.1007/s10009-002-0081-6.

Kaliszyk, C. 2007. "Web interfaces for proof assistants". *Electronic Notes in Theoretical Computer Science*. 174(2): 49–61. DOI: 10.1016/j.entcs.2006.09.021.

Kaliszyk, C., F. Chollet, and C. Szegedy. 2017a. "HolStep: A Machine Learning Dataset for Higher-order Logic Theorem Proving". In: *ICLR*. URL: https://openreview.net/forum?id=ryuxYmvel.

Kaliszyk, C. and R. O'Connor. 2009. "Computing with Classical Real Numbers". *Journal of Formalized Reasoning*. 2(1): 27–39. DOI: 10.6092/issn.1972-5787/1411.

Kaliszyk, C. and J. Urban. 2014. "Learning-Assisted Automated Reasoning with Flyspeck". *Journal of Automated Reasoning*. 53(2): 173–213. DOI: 10.1007/s10817-014-9303-3.

Kaliszyk, C. and J. Urban. 2015. "HOL(y)Hammer: Online ATP Service for HOL Light". *Mathematics in Computer Science*. 9(1): 5–22. DOI: 10.1007/s11786-014-0182-0.

Kaliszyk, C., J. Urban, and J. Vyskočil. 2017b. "Automating Formalization by Statistical and Semantic Parsing of Mathematics". In: *Interactive Theorem Proving*. Cham: Springer International Publishing. 12–27. DOI: 10.1007/978-3-319-66107-0_2.

Kammüller, F., M. Wenzel, and L. C. Paulson. 1999. "Locales A Sectioning Concept for Isabelle". In: *Theorem Proving in Higher Order Logics*. Berlin, Heidelberg: Springer. 149–165. DOI: 10.1007/3-540-48256-3_11.

Kästner, D., X. Leroy, S. Blazy, B. Schommer, M. Schmidt, and C. Ferdinand. 2017. "Closing the Gap – The Formally Verified Optimizing Compiler CompCert". In: *SSS'17: Safety-critical Systems Symposium 2017. Developments in System Safety Engineering: Proceedings of the Twenty-fifth Safety-critical Systems Symposium*. Bristol, UK: CreateSpace. 163–180. URL: https://hal.inria.fr/hal-01399482.

Kästner,
Schmidt, C. Ferdinand, X. Leroy, and S. Blazy. 2018. "CompCert: Practical experience on integrating and qualifying a formally verified optimizing compiler". In: *ERTS 2018: Embedded Real Time Software and Systems*. SEE. URL: https://hal.inria.fr/hal-01643290.

Kell, S., D. P. Mulligan, and P. Sewell. 2016. "The Missing Link: Explaining ELF Static Linking, Semantically". In: *OOPSLA*. Amsterdam, Netherlands: ACM. 607–623. DOI: 10.1145/2983990.2983996.

Kim, J., V. Sjöberg, R. Gu, and Z. Shao. 2017. "Safety and Liveness of MCS Lock - Layer by Layer". In: *APLAS*. Vol. 10695. *LNCS*. Springer. 273–297. DOI: 10.1007/978-3-319-71237-6_14.

Klein, G. 2014. "Proof Engineering Considered Essential". In: *FM 2014: Formal Methods: 19th International Symposium, Singapore, May 12-16, 2014. Proceedings*. Cham: Springer International Publishing. 16–21. DOI: 10.1007/978-3-319-06410-9_2.

Klein, G. 2015. "Gerwin's Style Guide for Isabelle/HOL". URL: https://proofcraft.org/blog/isabelle-style.html.

Klein, G., J. Andronick, K. Elphinstone, T. Murray, T. Sewell, R. Kolanski, and G. Heiser. 2014. "Comprehensive Formal Verification of an OS Microkernel". *ACM Trans. Comput. Syst.* 32(1): 2:1–2:70. DOI: 10.1145/2560537.

Klein, G., J. Andronick, M. Fernandez, I. Kuz, T. Murray, and G. Heiser. 2018. "Formally Verified Software in the Real World". *Commun. ACM*. 61(10): 68–77. DOI: 10.1145/3230627.

Klein, G., J. Andronick, G. Keller, D. Matichuk, T. Murray, and L. O'Connor. 2017. "Provably trustworthy systems". *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*. 375(2104). DOI: 10.1098/rsta.2015.0404.

Klein, G., K. Elphinstone, G. Heiser, J. Andronick, D. Cock, P. Derrin, D. Elkaduwe, K. Engelhardt, R. Kolanski, M. Norrish, T. Sewell, H. Tuch, and S. Winwood. 2009. "seL4: Formal Verification of an OS Kernel". In: *Proceedings of the ACM SIGOPS 22Nd Symposium on Operating Systems Principles. SOSP '09*. Big Sky, MT, USA: ACM. 207–220. DOI: 10.1145/1629575.1629596.

Klein, G., T. Nipkow, L. Paulson, and R. Thiemann. 2004-2019. “Archive of Formal Proofs: Submission Guidelines”. URL: https://www.isa-afp.org/submitting.html.

Ko, H.-S. and J. Gibbons. 2016. “Programming with ornaments”. *Journal of Functional Programming*. 27. DOI: 10.1017/S0956796816000307.

Kolbe, T. and C. Walther. 1998. “Proof analysis, generalization and reuse”. In: *Automated Deduction — A Basis for Applications*. Springer. 189–219. DOI: 10.1007/978-94-017-0435-9_8.

Komendantskaya, E., J. Heras, and G. Grov. 2012. “Machine Learning in Proof General: Interfacing Interfaces”. In: *Proceedings 10th International Workshop On User Interfaces for Theorem Provers, UITP 2012, Bremen, Germany, July 11th, 2012*. 15–41. DOI: 10.4204/EPTCS.118.2.

Krafft, D. B. 1981. “AVID: A system for the interactive development of verifiably correct programs”. *PhD thesis*. Cornell University.

Krebbers, R., J.-H. Jourdan, R. Jung, J. Tassarotti, J.-O. Kaiser, A. Timany, A. Charguéraud, and D. Dreyer. 2018. “MoSeL: A General, Extensible Modal Framework for Interactive Proofs in Separation Logic”. *Proc. ACM Program. Lang.* 2(ICFP): 77:1–77:30. DOI: 10.1145/3236772.

Krebbers, R., A. Timany, and L. Birkedal. 2017. “Interactive proofs in higher-order concurrent separation logic”. In: *POPL*. ACM. 205–217. DOI: 10.1145/3009837.3009855.

Kroening, D. and O. Strichman. 2008. *Decision Procedures: An Algorithmic Point of View*. 1st ed. Springer Publishing Company, Incorporated. DOI: 10.1007/978-3-540-74105-3.

Kühlwein, D., J. C. Blanchette, C. Kaliszyk, and J. Urban. 2013. “MaSh: Machine Learning for Sledgehammer”. In: *Interactive Theorem Proving*. Berlin, Heidelberg: Springer. 35–50. DOI: 10.1007/978-3-642-39634-2_6.

Kühlwein, D., T. van Laarhoven, E. Tsivtsivadze, J. Urban, and T. Heskes. 2012. “Overview and Evaluation of Premise Selection Techniques for Large Theory Mathematics”. In: *Automated Reasoning*. Berlin, Heidelberg: Springer. 378–392. DOI: 10.1007/978-3-642-31365-3_30.

Kumar, R. 2015. “Self-compilation and self-verification”. *PhD thesis*. University of Cambridge.

Kumar, R., M. O. Myreen, M. Norrish, and S. Owens. 2014. “CakeML: A Verified Implementation of ML”. In: *Proceedings of the 41st ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages. POPL ’14*. San Diego, CA, USA: ACM. 179–191. DOI: 10.1145/2535838.2535841.

Kumar, R., T. Kropf, and K. Schneider. 1991. “Integrating a first-order automatic prover in the HOL environment”. In: *1991 International Workshop on the HOL Theorem Proving System and Its Applications*. IEEE. 170–176. DOI: 10.1109/HOL.1991.596284.

Kunčar, O. and A. Popescu. 2018. “A Consistent Foundation for Isabelle/HOL”. *Journal of Automated Reasoning*. Jan. DOI: 10.1007/s10817-018-9454-8.

Lammich, P. 2013. “Automatic Data Refinement”. In: *Interactive Theorem Proving*. Berlin, Heidelberg: Springer. 84–99.

Lammich, P. 2015. “Refinement to imperative/HOL”. In: *International Conference on Interactive Theorem Proving*. Springer. 253–269.

Lampropoulos, L., Z. Paraskevopoulou, and B. C. Pierce. 2017. “Generating good generators for inductive relations”. *Proceedings of the ACM on Programming Languages*. 2(POPL): 45:1–45:30. DOI: 10.1145/3158133.

Lean Development Team. 2014-2017. “Theorem Proving in Lean”. URL: http://leanprover.github.io/tutorial/.

Lean Development Team. 2017-2018. “Javascript interface to the Lean server”. URL: http://github.com/leanprover/lean-client-js.

Lean Development Team. 2016-2019. “Lean for VS Code”. URL: http://github.com/leanprover/vscode-lean.

Leibniz, G. W. 1685. *The Art of Discovery*. Wiener 51.

Leino, K. R. M. 2010. “Dafny: An Automatic Program Verifier for Functional Correctness”. In: *Logic for Programming, Artificial Intelligence, and Reasoning*. Berlin, Heidelberg: Springer. 348–370. DOI: 10.1007/978-3-642-17511-4_20.

Lerner, S., S. R. Foster, and W. G. Griswold. 2015. “Polymorphic blocks: Formalism-inspired UI for structured connectors”. In: *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. ACM. 3063–3072. DOI: 10.1145/2702123.2702302.

Leroy, X. 2006. "Formal certification of a compiler back-end or: programming a compiler with a proof assistant". In: *POPL*. ACM. 42–54. DOI: 10.1145/1111037.1111042.

Leroy, X. 2007. "A locally nameless solution to the POPLMark challenge". *Research report* No. 6098. INRIA. URL: http://gallium.inria.fr/~xleroy/publi/POPLmark-locally-nameless.pdf.

Leroy, X. 2009. "Formal Verification of a Realistic Compiler". *Commun. ACM*. 52(7): 107–115. DOI: 10.1145/1538788.1538814.

Leroy, X. 2015. "Using Coq's evaluation mechanisms in anger". URL: http://gallium.inria.fr/blog/coq-eval/.

Leroy, X., A. W. Appel, S. Blazy, and G. Stewart. 2012. "The CompCert Memory Model, Version 2". *Research Report* No. RR-7987. INRIA. 26. URL: https://hal.inria.fr/hal-00703441.

Leroy, X. 2017. "The formal verification of compilers". URL: https://deepspec.org/event/dsss17/leroy-dsss17.pdf.

Lescuyer, S. and S. Conchon. 2009. "Improving Coq propositional reasoning using a lazy CNF conversion scheme". In: *International Symposium on Frontiers of Combining Systems*. Springer. 287–303.

Letouzey, P. 2003. "A New Extraction for Coq". In: *Types for Proofs and Programs*. Berlin, Heidelberg: Springer. 200–219. DOI: 10.1007/3-540-39185-1_12.

Letouzey, P. 2004. "Programmation fonctionnelle certifiee – L'extraction de programmes dans l'assistant Coq". *PhD thesis*. Universite Paris-Sud. URL: https://www.irif.fr/~letouzey/download/these_letouzey.pdf.

Letouzey, P. 2008. "Coq Extraction, an Overview". In: *Logic and Theory of Algorithms, Fourth Conference on Computability in Europe, CiE 2008*. Vol. 5028. Springer. DOI: 10.1007/978-3-540-69407-6_39.

Ley-Wild, R. and A. Nanevski. 2013. "Subjective auxiliary state for coarse-grained concurrency". In: *POPL*. ACM. 561–574. DOI: 10.1145/2429069.2429134.

Lin, Y., G. Grov, and R. Arthan. 2016. "Understanding and maintaining tactics graphically OR how we are learning that a diagram can be worth more than 10K LoC". *CoRR*. abs/1610.05593.

Lindblad, F. and M. Benke. 2006. "A Tool for Automated Theorem Proving in Agda". In: *Proceedings of the 2004 International Conference on Types for Proofs and Programs. TYPES'04*. Jouy-en-Josas, France: Springer-Verlag. 154–169. DOI: 10.1007/11617990_10.

Liu, Z., C. Morisset, and S. Wang. 2011. "A Graph-Based Implementation for Mechanized Refinement Calculus of OO Programs". In: *Formal Methods: Foundations and Applications*. Berlin, Heidelberg: Springer. 258–273. DOI: 10.1007/978-3-642-19829-8_17.

Loos, S., G. Irving, C. Szegedy, and C. Kaliszyk. 2017. "Deep Network Guided Proof Search". In: *LPAR-21. 21st International Conference on Logic for Programming, Artificial Intelligence and Reasoning*. Vol. 46. *EPiC Series in Computing*. EasyChair. 85–105. DOI: 10.29007/8mwc.

Luo, Z. 1999. "Coercive subtyping". *Journal of Logic and Computation*. 9(1): 105–130. DOI: 10.1093/logcom/9.1.105.

Lynch, N. and F. Vaandrager. 1994. "FORWARD AND BACKWARD SIMULATIONS PART I: UNTIMED SYSTEMS (Replaces TM-486)". *Tech. rep.* Cambridge, MA, USA.

MacQueen, D. B. 1986. "Using Dependent Types to Express Modular Structure". In: *Proceedings of the 13th ACM SIGACT-SIGPLAN Symposium on Principles of Programming Languages. POPL '86*. St. Petersburg Beach, Florida: ACM. 277–286. DOI: 10.1145/512644.512670.

Magaud, N. and Y. Bertot. 2002. "Changing Data Structures in Type Theory: A Study of Natural Numbers". In: *Types for Proofs and Programs: International Workshop. TYPES 2000*. Berlin, Heidelberg: Springer. 181–196. DOI: 10.1007/3-540-45842-5_12.

Mahboubi, A. and E. Tassi. 2013. "Canonical Structures for the Working Coq User". In: *Interactive Theorem Proving*. Vol. 7998. *LNCS*. Berlin, Heidelberg: Springer. 19–34. DOI: 10.1007/978-3-642-39634-2_5.

Malecha, G. and J. Bengtson. 2016. “Extensible and Efficient Automation Through Reflective Tactics”. In: *Programming Languages and Systems: 25th European Symposium on Programming, ESOP 2016, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2016, Eindhoven, The Netherlands, April 2–8, 2016, Proceedings*. Berlin, Heidelberg: Springer. 532–559. DOI: 10.1007/978-3-662-49498-1_21.

Malecha, G., G. Morrisett, A. Shinnar, and R. Wisnesky. 2010. “Toward a Verified Relational Database Management System”. In: *Proceedings of the 37th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages. POPL ’10*. Madrid, Spain: ACM. 237–248. DOI: 10.1145/1706299.1706329.

Martin-Löf, P. 1982. “Constructive Mathematics and Computer Programming”. In: *Logic, Methodology and Philosophy of Science VI*. Vol. 104. *Studies in Logic and the Foundations of Mathematics*. Elsevier. 153–175. DOI: 10.1016/S0049-237X(09)70189-2.

Martin-Löf, P. 1984. *Intuitionistic Type Theory*. Bibliopolis.

Matichuk, D., T. Murray, J. Andronick, R. Jeffery, G. Klein, and M. Staples. 2015a. “Empirical Study Towards a Leading Indicator for Cost of Formal Software Verification”. In: *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering*. Vol. 1. 722–732. DOI: 10.1109/ICSE.2015.85.

Matichuk, D., T. C. Murray, and M. Wenzel. 2015b. “Eisbach: A Proof Method Language for Isabelle”. *Journal of Automated Reasoning*. 56: 261–282. DOI: 10.1007/s10817-015-9360-2.

Matichuk, D., M. Wenzel, and T. Murray. 2015c. “The Eisbach user manual”. *Isabelle Community*.

McBride, C. 1996. “Inverting inductively defined relations in LEGO”. In: *International Workshop on Types for Proofs and Programs*. Springer. 236–253.

McBride, C. 2002. “Elimination with a Motive”. In: *Types for Proofs and Programs*. Berlin, Heidelberg: Springer. 197–216. DOI: 10.1007/3-540-45842-5_13.

McBride, C. 2011. “Ornamental algebras, algebraic ornaments”. URL: http://plv.mpi-sws.org/plerg/papers/mcbride-ornaments-2up.pdf.

McBride, C. 2015. “Turing-Completeness Totally Free”. In: *Mathematics of Program Construction*. Cham: Springer International Publishing. 257–275. DOI: 10.1007/978-3-319-19797-5_13.

McCarthy, J. 1960. “Recursive Functions of Symbolic Expressions and Their Computation by Machine, Part I”. *Commun. ACM*. 3(4): 184–195. DOI: 10.1145/367177.367199.

McCarthy, J. 1963. “A Basis for a Mathematical Theory of Computation”. In: *Computer Programming and Formal Systems*. Ed. by P. Braffort and D. Hirschberg. Vol. 35. *Studies in Logic and the Foundations of Mathematics*. Elsevier. 33–70. DOI: 10.1016/S0049-237X(08)72018-4.

McCreight, A. 2009. “Practical tactics for separation logic”. In: *International Conference on Theorem Proving in Higher Order Logics*. Springer. 343–358. DOI: 10.1007/978-3-642-03359-9_24.

Mehnert, H. and D. Christiansen. 2014. “Tool Demonstration: An IDE for Programming and Proving in Idris”. *Proceedings of Vienna Summer of Logic, VSL*. 14: 2.

Mehta, F. and T. Nipkow. 2003. “Proving Pointer Programs in Higher-Order Logic”. In: *CADE*. Vol. 2741. *LNCS*. Springer. 121–135. DOI: 10.1007/978-3-540-45085-6_10.

Microsoft. 1997-2019. “Visual Studio”. URL: http://visualstudio.microsoft.com/.

Miller, D. 2018. “Mechanized Metatheory Revisited”. *Journal of Automated Reasoning*. Oct. DOI: 10.1007/s10817-018-9483-3.

Milner, R. 1972. “Implementation and Applications of Scott’s Logic for Computable Functions”. In: *Proceedings of ACM Conference on Proving Assertions About Programs*. Las Cruces, NM, USA: ACM. 1–6. DOI: 10.1145/800235.807067.

Milner, R., M. Tofte, R. Harper, and D. MacQueen. 1997. *The Definition of Standard ML (Revised)*. Cambridge, MA, USA: MIT Press.

Milner, R. and R. Weyhrauch. 1972. “Proving compiler correctness in a mechanized logic”. *Machine Intelligence*. 7: 51–70.

Monperrus, M. 2018. “Automatic Software Repair: A Bibliography”. *ACM Comput. Surv.* 51(1): 17:1–17:24. DOI: 10.1145/3105906.

Morrisett, G., G. Tan, J. Tassarotti, J.-B. Tristan, and E. Gan. 2012. "RockSalt: Better, Faster, Stronger SFI for the x86". In: *Proceedings of the 33rd ACM SIGPLAN Conference on Programming Language Design and Implementation. PLDI '12*. Beijing, China: ACM. 395–404. DOI: 10.1145/2254064.2254111.

Mulhern, A. 2006. "Proof Weaving". In: *In Proceedings of the First Informal ACM SIGPLAN Workshop on Mechanizing Metatheory*.

Mullen, E., S. Pernsteiner, J. R. Wilcox, Z. Tatlock, and D. Grossman. 2018. "Oeuf: Minimizing the Coq Extraction TCB". In: *CPP*. Los Angeles, CA, USA: ACM. 172–185. DOI: 10.1145/3167089.

Müller, D., T. Gauthier, C. Kaliszyk, M. Kohlhase, and F. Rabe. 2017. "Classification of Alignments Between Concepts of Formal Mathematical Systems". In: *Intelligent Computer Mathematics*. Cham: Springer International Publishing. 83–98. DOI: 10.1007/978-3-319-62075-6_7.

Mulligan, D. P., S. Owens, K. E. Gray, T. Ridge, and P. Sewell. 2014. "Lem: Reusable Engineering of Real-world Semantics". In: *Proceedings of the 19th ACM SIGPLAN International Conference on Functional Programming. ICFP '14*. Gothenburg, Sweden: ACM. 175–188. DOI: 10.1145/2628136.2628143.

Murphy-Hill, E. and D. Grossman. 2014. "How Programming Languages Will Co-evolve with Software Engineering: A Bright Decade Ahead". In: *Proceedings of the on Future of Software Engineering. FOSE 2014*. Hyderabad, India: ACM. 145–154. DOI: 10.1145/2593882.2593898.

Murray, T. and P. C. van Oorschot. 2018. "BP: Formal Proofs, the Fine Print and Side Effects". In: *IEEE Cybersecurity Development (SecDev)*. 1–10. DOI: 10.1109/SecDev.2018.00009.

Myreen, M. O. 2008-2018. "Guide to HOL4 interaction and basic proofs". URL: http://hol-theorem-prover.org/HOL-interaction.pdf.

Myreen, M. O. and S. Owens. 2012. "Proof-producing Synthesis of ML from Higher-order Logic". In: *Proceedings of the 17th ACM SIGPLAN International Conference on Functional Programming. ICFP '12*. Copenhagen, Denmark: ACM. 115–126. DOI: 10.1145/2364527.2364545.

Nagashima, Y. and Y. He. 2018. “PaMpeR: Proof Method Recommendation System for Isabelle/HOL”. In: *Proceedings of the International Conference on Automated Software Engineering. ASE 2018*. Montpellier, France: ACM. 362–372. DOI: 10.1145/3238147.3238210.

Nagashima, Y. and R. Kumar. 2017. “A Proof Strategy Language and Proof Script Generation for Isabelle/HOL”. In: *Automated Deduction – CADE 26*. Cham: Springer International Publishing. 528–545. DOI: 10.1007/978-3-319-63046-5_32.

Nanevski, A., R. Ley-Wild, I. Sergey, and G. A. Delbianco. 2014. “Communicating State Transition Systems for Fine-Grained Concurrent Resources”. In: *Programming Languages and Systems*. Ed. by Z. Shao. Berlin, Heidelberg: Springer. 290–310. DOI: 10.1007/978-3-642-54833-8_16.

Nanevski, A., G. Morrisett, A. Shinnar, P. Govereau, and L. Birkedal. 2008a. “Ynot: Dependent Types for Imperative Programs”. In: *Proceedings of the 13th ACM SIGPLAN International Conference on Functional Programming. ICFP ’08*. Victoria, BC, Canada: ACM. 229–240. DOI: 10.1145/1411204.1411237.

Nanevski, A., F. Pfenning, and B. Pientka. 2008b. “Contextual modal type theory”. *ACM Transactions on Computational Logic*. 9(3): 23. DOI: 10.1145/1352582.1352591.

Nanevski, A., V. Vafeiadis, and J. Berdine. 2010. “Structuring the Verification of Heap-manipulating Programs”. In: *Proceedings of the 37th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages. POPL ’10*. Madrid, Spain: ACM. 261–274. DOI: 10.1145/1706299.1706331.

Narboux, J. 2004. “A decision procedure for geometry in Coq”. In: *International Conference on Theorem Proving in Higher Order Logics*. Springer. 225–240. DOI: 10.1007/978-3-540-30142-4_17.

Neis, G., C.-K. Hur, J.-O. Kaiser, C. McLaughlin, D. Dreyer, and V. Vafeiadis. 2015. “Pilsner: A Compositionally Verified Compiler for a Higher-order Imperative Language”. In: *Proceedings of the 20th ACM SIGPLAN International Conference on Functional Programming. ICFP 2015*. Vancouver, BC, Canada: ACM. 166–178. DOI: 10.1145/2784731.2784764.

Newey, M. C. 1973. “Axioms and Theorems for Integers, Lists and Finite Sets in LCF”. *Tech. rep.* Memo AIM-184. Stanford, CA, USA.

Nipkow, T. 1989. “Term rewriting and beyond–theorem proving in Isabelle”. *Formal Aspects of Computing*. 1(1): 320–338. DOI: 10.1007/BF01887212.

Nipkow, T. 1990. “Proof transformations for equational theories”. In: *Symposium on Logic in Computer Science*. IEEE. 278–288. DOI: 10.1109/LICS.1990.113754.

Nipkow, T. and G. Klein. 2014. *Concrete Semantics: With Isabelle/HOL*. Springer. DOI: 10.1007/978-3-319-10542-0.

nLab authors. 2019a. “equality”. URL: http://ncatlab.org/nlab/show/equality.

nLab authors. 2019b. “foundation of mathematics”. URL: http://ncatlab.org/nlab/show/foundation%20of%20mathematics.

nLab authors. 2019c. “intensional type theory”. URL: http://ncatlab.org/nlab/show/intensional%20type%20theory.

Norell, U. 2016. “Agda reflection overhaul”. URL: http://lists.chalmers.se/pipermail/agda/2016/008414.html.

Norell, U. 2015-2019. “agda-prelude”. URL: http://github.com/UlfNorell/agda-prelude.

O’Connor, L., C. Rizkallah, Z. Chen, S. Amani, J. Lim, Y. Nagashima, T. Sewell, A. Hixon, G. Keller, T. C. Murray, and G. Klein. 2016. “COGENT: Certified Compilation for a Functional Systems Language”. *CoRR*. abs/1601.05520.

O’Connor, R. 2005. “Essential Incompleteness of Arithmetic Verified by Coq”. In: *Theorem Proving in Higher Order Logics: 18th International Conference, TPHOLs 2005, Oxford, UK, August 22-25, 2005. Proceedings*. Berlin, Heidelberg: Springer. 245–260. DOI: 10.1007/11541868_16.

O’Hearn, P. W. 2007. “Resources, concurrency, and local reasoning”. *Theor. Comput. Sci.* 375(1-3): 271–307. DOI: 10.1016/j.tcs.2006.12.035.

O’Hearn, P. W., J. C. Reynolds, and H. Yang. 2001. “Local Reasoning about Programs that Alter Data Structures”. In: *CSL*. Vol. 2142. *LNCS*. Springer. 1–19. DOI: 10.1007/3-540-44802-0_1.

Opdyke, W. F. 1992. “Refactoring: A program restructuring aid in designing object-oriented application frameworks”. *PhD thesis*. University of Illinois at Urbana-Champaign.

Oury, N. 2005. “Extensionality in the Calculus of Constructions”. In: *Theorem Proving in Higher Order Logics*. Berlin, Heidelberg: Springer. 278–293. DOI: 10.1007/11541868_18.

Owens, S. 2008. “A sound semantics for OCaml Light”. *Programming Languages and Systems*: 1–15. DOI: 10.1007/978-3-540-78739-6_1.

Palmskog, K., A. Celik, and M. Gligoric. 2018. “piCoq: Parallel Regression Proving for Large-scale Verification Projects”. In: *ISSTA*. Amsterdam, Netherlands: ACM. 344–355. DOI: 10.1145/3213846.3213877.

Paraskevopoulou, Z., C. Hriţcu, M. Dénès, L. Lampropoulos, and B. C. Pierce. 2015. “Foundational Property-Based Testing”. In: *Interactive Theorem Proving: 6th International Conference, ITP 2015, Nanjing, China, August 24-27, 2015, Proceedings*. Cham: Springer International Publishing. 325–343. DOI: 10.1007/978-3-319-22102-1_22.

Paulin-Mohring, C. 1989a. “Extracting F-omega’s Programs from Proofs in the Calculus of Constructions”. In: *Proceedings of the 16th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages. POPL ’89*. Austin, TX, USA: ACM. 89–104. DOI: 10.1145/75277.75285.

Paulin-Mohring, C. 1989b. “Extraction de programmes dans le Calcul des Constructions”. *PhD thesis*. Université Paris 7.

Paulin-Mohring, C. 1993. “Inductive definitions in the system Coq rules and properties”. In: *Typed Lambda Calculi and Applications*. Berlin, Heidelberg: Springer. 328–345. DOI: 10.1007/BFb0037116.

Paulin-Mohring, C. and B. Werner. 1993. “Synthesis of ML programs in the system Coq”. *Journal of Symbolic Computation*. 15(5): 607–640. DOI: 10.1016/S0747-7171(06)80007-6.

Paulson, L. 1983. “Tactics and tacticals in Cambridge LCF”. *Tech. rep.* University of Cambridge, Computer Laboratory.

Paulson, L. 1984. “Deriving structural induction in LCF”. In: *Semantics of Data Types*. Berlin, Heidelberg: Springer. 197–214. DOI: 10.1007/3-540-13346-1_10.

Paulson, L. C. 1988. “A preliminary users manual for Isabelle”. *Tech. rep.* University of Cambridge, Computer Laboratory.

Paulson, L. C. 1993. “Isabelle: The Next 700 Theorem Provers”. *CoRR*. cs.LO/9301106.

Paulson, L. C. 1994. “Isabelle: A Generic Theorem Prover”. In: *Lecture Notes in Computer Science*. Vol. 828. DOI: 10.1007/BFb0030541.

Paulson, L. C. 1997. “Mechanizing Coinduction and Corecursion in Higher-order Logic”. *Journal of Logic and Computation*. 7(2): 175–204. DOI: 10.1093/logcom/7.2.175.

Paulson, L. C. 1999. “A generic tableau prover and its integration with Isabelle”. *Journal of Universal Computer Science*. 5(3): 73–87. DOI: 10.3217/jucs-005-03-0073.

Paulson, L. C. 2006. “Defining Functions on Equivalence Classes”. *ACM Trans. Comput. Logic*. 7(4): 658–675. DOI: 10.1145/1183278.1183280.

Paulson, L. C. and J. C. Blanchette. 2012. “Three years of experience with Sledgehammer, a practical link between automatic and interactive theorem provers”. In: *International Workshop on the Implementation of Logics (IWIL 2010)*. Vol. 2. *EPiC Series*. EasyChair. 1–11.

Pédrot, P.-M. 2019. “Ltac2: Tactical Warfare”. In: *CoqPL 2019*.

Peng, K. and D. Ma. 2017. “Tree-Structure CNN for Automated Theorem Proving”. In: *Neural Information Processing*. Cham: Springer International Publishing. 3–12. DOI: 10.1007/978-3-319-70096-0_1.

Pfenning, F. and C. Elliott. 1988. “Higher-order Abstract Syntax”. In: *Proceedings of the ACM SIGPLAN 1988 Conference on Programming Language Design and Implementation. PLDI ’88*. Atlanta, GA, USA: ACM. 199–208. DOI: 10.1145/53990.54010.

Pfenning, F. 2010. “Lecture Notes on Proofs as Programs: Lecture 2”. URL: http://www.cs.cmu.edu/~fp/courses/15816-s10/lectures/02-pap.pdf.

Pfenning, F. and C. Paulin-Mohring. 1990. “Inductively defined types in the Calculus of Constructions”. In: *Mathematical Foundations of Programming Semantics*. New York, NY: Springer-Verlag. 209–228. DOI: 10.1007/BFb0040259.

Pfenning, F. and C. Schürmann. 1999. "System Description: Twelf — A Meta-Logical Framework for Deductive Systems". In: *Automated Deduction — CADE-16: 16th International Conference on Automated Deduction Trento, Italy, July 7–10, 1999 Proceedings*. Berlin, Heidelberg: Springer. 202–206. DOI: 10.1007/3-540-48660-7_14.

PG development team. 2016. "Proof General 4.4.1 pre Documentation". URL: https://proofgeneral.github.io/doc/userman/.

Pientka, B. and J. Dunfield. 2008. "Programming with proofs and explicit contexts". In: *Proceedings of the 10th international ACM SIGPLAN conference on Principles and practice of declarative programming*. ACM. 163–173. DOI: 10.1145/1389449.1389469.

Pierce, B. C. 2002. *Types and programming languages*. MIT press.

Pierce, B. C. 2017. Personal communication.

Pierce, B. C. 2019. Personal communication.

Pierce, B. C., C. Casinghino, M. Gaboardi, M. Greenberg, C. Hriţcu, V. Sjöberg, and B. Yorgey. 2014. *Software Foundations*. Electronic textbook. URL: http://www.cis.upenn.edu/~bcpierce/sf.

Pit-Claudel, C. and P. Courtieu. 2016. "Company-Coq: Taking Proof General one step closer to a real IDE". In: *CoqPL'16: The Second International Workshop on Coq for PL*. DOI: 10.5281/zenodo.44331.

Podkopaev, A., O. Lahav, and V. Vafeiadis. 2019. "Bridging the gap between programming languages and hardware weak memory models". *PACMPL*. 3(POPL): 69:1–69:31. DOI: 10.1145/3290382.

Pollack, R. 1995. "On extensibility of proof checkers". In: *Types for Proofs and Programs*. Ed. by P. Dybjer, B. Nordström, and J. Smith. Berlin, Heidelberg: Springer. 140–161. DOI: 10.1007/3-540-60579-7_8.

Pollack, R. 1998. "How to Believe a Machine-Checked Proof". In: *Twenty Five Years of Constructive Type Theory*. Ed. by G. Sambin and J. Smith. Oxford University Press.

Pons, O. 1999. "Conception et rÉalisation d'outils d'aide au dÉveloppement de grosses thÉories dans les systèmes de preuves interactifs". *PhD thesis*.

Poswolsky, A. and C. Schürmann. 2009. "System description: Delphin–a functional programming language for deductive systems". *Electronic Notes in Theoretical Computer Science*. 228: 113–120.

Pouillard, N. 2012. "agda-tactics: Semiring". URL: http://github.com/xplat/agda-tactics/blob/master/Tactics/Nat/Semiring.agda.

Pugh, W. 1991. "The Omega Test: A Fast and Practical Integer Programming Algorithm for Dependence Analysis". In: *Proceedings of the 1991 ACM/IEEE Conference on Supercomputing. Supercomputing '91.* Albuquerque, NM, USA: ACM. 4–13. DOI: 10.1145/125826.125848.

Qi, Z., F. Long, S. Achour, and M. Rinard. 2015. "An Analysis of Patch Plausibility and Correctness for Generate-and-validate Patch Generation Systems". In: *Proceedings of the 2015 International Symposium on Software Testing and Analysis. ISSTA 2015.* Baltimore, MD, USA: ACM. 24–36. DOI: 10.1145/2771783.2771791.

Rand, R., J. Paykin, and S. Zdancewic. 2017. "QWIRE Practice: Formal Verification of Quantum Circuits in Coq". In: *Proceedings 14th International Conference on Quantum Physics and Logic, QPL 2017, Nijmegen, The Netherlands, 3-7 July 2017.* 119–132. DOI: 10.4204/EPTCS.266.8.

RedPRL Development Team. 2015-2018. "The RedPRL Proof Assistant". URL: http://www.redprl.org.

Reynolds, J. C. 2002. "Separation Logic: A Logic for Shared Mutable Data Structures". In: *LICS.* IEEE Computer Society. 55–74. DOI: 10.1109/LICS.2002.1029817.

Ricketts, D., V. Robert, D. Jang, Z. Tatlock, and S. Lerner. 2014. "Automating Formal Proofs for Reactive Systems". In: *Proceedings of the 35th ACM SIGPLAN Conference on Programming Language Design and Implementation. PLDI '14.* Edinburgh, UK: ACM. 452–462. DOI: 10.1145/2594291.2594338.

Ridge, T., D. Sheets, T. Tuerk, A. Giugliano, A. Madhavapeddy, and P. Sewell. 2015. "SibylFS: Formal Specification and Oracle-based Testing for POSIX and Real-world File Systems". In: *Proceedings of the 25th Symposium on Operating Systems Principles. SOSP '15.* Monterey, California: ACM. 38–53. DOI: 10.1145/2815400.2815411.

Ringer, T. and N. Yazdani. 2018. "PUMPKIN-git". URL: http://github.com/uwplse/PUMPKIN-git.

Full text available at: http://dx.doi.org/10.1561/2500000045

Ringer, T., N. Yazdani, J. Leo, and D. Grossman. 2018. "Adapting Proof Automation to Adapt Proofs". In: *Proceedings of the 7th ACM SIGPLAN Conference on Certified Programs and Proofs. CPP 2018.* DOI: 10.1145/3167094.

Ringer, T., N. Yazdani, J. Leo, and D. Grossman. 2019. "Ornaments for Proof Reuse in Coq". In: *Interactive Theorem Proving.*

Rizkallah, C., J. Lim, Y. Nagashima, T. Sewell, Z. Chen, L. O'Connor, T. Murray, G. Keller, and G. Klein. 2016. "A Framework for the Automatic Formal Verification of Refinement from Cogent to C". In: *International Conference on Interactive Theorem Proving.* Nancy, France. DOI: 10.1007/978-3-319-43144-4_20.

Robert, V. 2018. "Front-end tooling for building and maintaining dependently-typed functional programs". *PhD thesis.* UC San Diego.

Robert, V. and S. Lerner. 2014-2016. "PeaCoq". URL: http://goto.ucsd.edu/peacoq/.

Robinson, J. A. 1965. "A Machine-Oriented Logic Based on the Resolution Principle". *J. ACM.* 12(1): 23–41. DOI: 10.1145/321250.321253.

Roe, K. and S. Smith. 2016. "CoqPIE: An IDE Aimed at Improving Proof Development Productivity". In: *Interactive Theorem Proving: 7th International Conference, ITP 2016, Nancy, France, August 22-25, 2016, Proceedings.* Cham: Springer International Publishing. 491–499. DOI: 10.1007/978-3-319-43144-4_32.

Rompf, T. and N. Amin. 2016. "Type Soundness for Dependent Object Types". In: *Proceedings of the 2016 ACM SIGPLAN International Conference on Object-Oriented Programming, Systems, Languages, and Applications. OOPSLA 2016.* Amsterdam, Netherlands: ACM. 624–641. DOI: 10.1145/2983990.2984008.

Rushby, J. M. 1981. "Design and Verification of Secure Systems". In: *Proceedings of the Eighth ACM Symposium on Operating Systems Principles. SOSP '81.* Pacific Grove, CA, USA: ACM. 12–21. DOI: 10.1145/800216.806586.

Russell, B. 1906. "On Some Difficulties in the Theory of Transfinite Numbers and Order Types". *Proceedings of the London Mathematical Society.* s2-4(1): 29–53. DOI: 10.1112/plms/s2-4.1.29.

Russell, B. 1918. *Introduction to Mathematical Philosophy.* George Allen and Unwin.

Saibi, A. 1997. "Typing Algorithm in Type Theory with Inheritance". In: *Proceedings of the 24th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages. POPL '97*. Paris, France: ACM. 292–301. DOI: 10.1145/263699.263742.

Saibi, A. 1999. "Outils Génériques de Modélisation et de Démonstration pour la Formalisation des Mathématiques en Théorie des Types: application à la Théorie des Catégories". *PhD thesis*. Paris, France: Université Paris VI.

Sangiorgi, D. 2011. *Introduction to Bisimulation and Coinduction*. New York, NY, USA: Cambridge University Press. DOI: 10.1017/CBO9780511777110.

Schäfer, S., T. Tebbi, and G. Smolka. 2015. "Autosubst: Reasoning with de Bruijn terms and parallel substitutions". In: *International Conference on Interactive Theorem Proving*. Springer. 359–374. DOI: 10.1007/978-3-319-22102-1_24.

Schlichtkrull, A., J. C. Blanchette, and D. Traytel. 2019. "A verified prover based on ordered resolution". In: *Proceedings of the 8th ACM SIGPLAN International Conference on Certified Programs and Proofs*. ACM. 152–165.

Scott, D. 1970. "Constructive validity". In: *Symposium on Automatic Demonstration*. Berlin, Heidelberg: Springer. 237–275. DOI: 10.1007/BFb0060636.

Scott, D. S. 1993. "A type-theoretical alternative to ISWIM, CUCH, OWHY". *Theoretical Computer Science*. 121(1): 411–440. DOI: 10.1016/0304-3975(93)90095-B.

Selsam, D. and L. de Moura. 2017. "Congruence Closure in Intensional Type Theory". *CoRR*. abs/1701.04391.

Selsam, D., P. Liang, and D. L. Dill. 2017. "Developing Bug-free Machine Learning Systems with Formal Mathematics". In: *Proceedings of the 34th International Conference on Machine Learning - Volume 70. ICML'17*. Sydney, NSW, Australia: JMLR.org. 3047–3056.

Sergey, I., A. Nanevski, and A. Banerjee. 2015. "Mechanized Verification of Fine-grained Concurrent Programs". In: *Proceedings of the 36th ACM SIGPLAN Conference on Programming Language Design and Implementation. PLDI '15*. Portland, OR, USA: ACM. 77–87. DOI: 10.1145/2737924.2737964.

Sergey, I., J. R. Wilcox, and Z. Tatlock. 2017. “Programming and Proving with Distributed Protocols”. *Proc. ACM Program. Lang.* 2(POPL): 28:1–28:30. DOI: 10.1145/3158116.

Sewell, P., F. Zappa Nardelli, S. Owens, G. Peskine, T. Ridge, S. Sarkar, and R. StrniŠa. 2010. “Ott: Effective tool support for the working semanticist”. *Journal of Functional Programming*. 20(1): 71–122. DOI: 10.1017/S0956796809990293.

Sewell, T., F. Kam, and G. Heiser. 2017. “High-assurance timing analysis for a high-assurance real-time operating system”. *Real-Time Systems*. 53(5): 812–853. DOI: 10.1007/s11241-017-9286-3.

Shah, N. 2005. “"Rippling: Meta-Level Guidance for Mathematical Reasoning, " by Alan Bundy, David Basin, Dieter Hutter, and Andrew Ireland, Cambridge University Press, 2005”. *J. Autom. Reasoning*. 35(4): 429–431. DOI: 10.1007/s10817-006-9027-0.

Shaw, M., J. Aldrich, T. D. Breaux, D. Garlan, C. Kästner, C. Le Goues, and W. L. Scherlis. 2015. “Seminal Papers in Software Engineering: The Carnegie Mellon Canonical Collection”. URL: http://reports-archive.adm.cs.cmu.edu/anon/isr2015/CMU-ISR-15-107.pdf.

Slind, K. 1994. “AC unification in HOL90”. In: *Higher Order Logic Theorem Proving and Its Applications*. Springer. 437–449. DOI: 10.1007/3-540-57826-9_154.

Slotosch, O. 1997. “Higher order quotients and their implementation in Isabelle/HOL”. In: *Theorem Proving in Higher Order Logics*. Berlin, Heidelberg: Springer. 291–306. DOI: 10.1007/BFb0028401.

Smith, R. 2018. “Aristotle’s Logic”. In: *The Stanford Encyclopedia of Philosophy*. Ed. by E. N. Zalta. Spring 2018. Metaphysics Research Lab, Stanford University. URL: https://plato.stanford.edu/archives/spr2018/entries/aristotle-logic/.

Sørensen, M. H. and P. Urzyczyn. 2006. *Lectures on the Curry-Howard isomorphism*. Vol. 149. Elsevier.

Sozeau, M. 2010. “Equations: A Dependent Pattern-Matching Compiler”. In: *Interactive Theorem Proving*. Berlin, Heidelberg: Springer. 419–434. DOI: 10.1007/978-3-642-14052-5_29.

Sozeau, M., A. Anand, S. Boulier, C. Cohen, Y. Forster, F. Kunze, G. Malecha, N. Tabareau, and T. Winterhalter. 2019. “The MetaCoq Project”. *Tech. rep.* INRIA. URL: https://hal.inria.fr/hal-02167423.

Sozeau, M. and N. Oury. 2008. "First-Class Type Classes". In: *Theorem Proving in Higher Order Logics: 21st International Conference, TPHOLs 2008, Montreal, Canada, August 18-21, 2008. Proceedings.* Berlin, Heidelberg: Springer. 278–293. DOI: 10.1007/978-3-540-71067-7_23.

Sozeau, M. and N. Tabareau. 2014. "Universe Polymorphism in Coq". In: *Interactive Theorem Proving.* Cham: Springer International Publishing. 499–514. DOI: 10.1007/978-3-319-08970-6_32.

Spitters, B. and E. van der Weegen. 2011. "Type classes for mathematics in type theory". *Mathematical Structures in Computer Science.* 21(4): 795–825. DOI: 10.1017/S0960129511000119.

Spiwack, A. 2010. "An abstract type for constructing tactics in Coq". In: *Proof Search in Type Theory.* Edinburgh, UK. URL: https://hal.inria.fr/inria-00502500.

Spiwack, A. 2016. "Inside the design of a tactic system". URL: https://github.com/aspiwack/tacengine/releases/download/v0.1-draft/tacengine.pdf.

Stampoulis, A. and Z. Shao. 2010. "VeriML: Typed Computation of Logical Terms Inside a Language with Effects". In: *Proceedings of the 15th ACM SIGPLAN International Conference on Functional Programming. ICFP '10.* Baltimore, MD, USA: ACM. 333–344. DOI: 10.1145/1863543.1863591.

Staples, M., R. Jeffery, J. Andronick, T. Murray, G. Klein, and R. Kolanski. 2014. "Productivity for Proof Engineering". In: *Proceedings of the 8th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement. ESEM '14.* Torino, Italy: ACM. 15:1–15:4. DOI: 10.1145/2652524.2652551.

Staples, M., R. Kolanski, G. Klein, C. Lewis, J. Andronick, T. Murray, R. Jeffery, and L. Bass. 2013. "Formal Specifications Better Than Function Points for Code Sizing". In: *Proceedings of the 2013 International Conference on Software Engineering. ICSE '13.* San Francisco, CA, USA: IEEE Press. 1257–1260. DOI: 10.1109/ICSE.2013.6606692.

Sterling, J. and R. Harper. 2017. "Algebraic Foundations of Proof Refinement". *CoRR.* abs/1703.05215.

Stewart, G., L. Beringer, S. Cuellar, and A. W. Appel. 2015. "Compositional CompCert". In: *POPL*. Mumbai, India: ACM. 275–287. DOI: 10.1145/2676726.2676985.

Strachey, C. 2000. "Fundamental Concepts in Programming Languages". *Higher-Order and Symbolic Computation*. 13(1): 11–49. DOI: 10.1023/A:1010000313106.

StructTact Development Team. 2016-2019. "StructTact". URL: http://github.com/uwplse/StructTact.

Stump, A. 2017. "The calculus of dependent lambda eliminations". *Journal of Functional Programming*. 27. DOI: 10.1017/S0956796817000053.

Svendsen, K. and L. Birkedal. 2014. "Impredicative Concurrent Abstract Predicates". In: 149–168.

Swamy, N., C. Hriţcu, C. Keller, A. Rastogi, A. Delignat-Lavaud, S. Forest, K. Bhargavan, C. Fournet, P.-Y. Strub, M. Kohlweiss, J.-K. Zinzindohoue, and S. Zanella-Béguelin. 2016. "Dependent Types and Multi-monadic Effects in F*". *SIGPLAN Not.* 51(1): 256–270. DOI: 10.1145/2914770.2837655.

Swamy, N., J. Weinberger, C. Schlesinger, J. Chen, and B. Livshits. 2013. "Verifying higher-order programs with the Dijkstra monad". In: *PLDI*. ACM. 387–398. DOI: 10.1145/2491956.2491978.

Swierstra, W. 2008. "Data Types à La Carte". *J. Funct. Program.* 18(4): 423–436. DOI: 10.1017/S0956796808006758.

Swierstra, W. 2009. "A Hoare Logic for the State Monad". In: *TPHOLs*. Vol. 5674. *LNCS*. Springer. 440–451. DOI: 10.1007/978-3-642-03359-9_30.

Syme, D. 1995. "A new interface for HOL — Ideas, issues and implementation". In: *Higher Order Logic Theorem Proving and Its Applications*. Berlin, Heidelberg: Springer. 324–339. DOI: 10.1007/3-540-60275-5_74.

Tabareau, N., É. Tanter, and M. Sozeau. 2018. "Equivalences for Free: Univalent Parametricity for Effective Transport". *Proc. ACM Program. Lang.* 2(ICFP): 92:1–92:29. DOI: 10.1145/3236787.

Tanter, É. and N. Tabareau. 2015. "Gradual Certified Programming in Coq". In: *Proceedings of the 11th Symposium on Dynamic Languages. DLS 2015.* Pittsburgh, PA, USA: ACM. 26–40. DOI: 10.1145/2816707.2816710.

Tarski, A. 1936. "Der Wahrheitsbegriff in den Formalisierten Sprachen". *Studia Philosophica*. 1: 261–405.

Trybulec, A. and H. A. Blair. 1985. "Computer Assisted Reasoning with Mizar". In: *IJCAI*. Vol. 85. Citeseer. 26–28.

Turing, A. 1949. "Checking a large routine". In: *Report of a Conference on High Speed Automatic Calculating Machines*. 67–69.

Univalent Foundations Program. 2013. *Homotopy Type Theory: Univalent Foundations of Mathematics*. Institute for Advanced Study. URL: https://homotopytypetheory.org/book.

Urban, C. 2008. "Nominal Techniques in Isabelle/HOL". *Journal of Automated Reasoning*. 40(4): 327–356. DOI: 10.1007/s10817-008-9097-2.

Urban, C. and C. Kaliszyk. 2011. "General Bindings and Alpha-Equivalence in Nominal Isabelle". In: *Programming Languages and Systems*. Berlin, Heidelberg: Springer. 480–500.

Valbuena, I. L. and M. Johansson. 2015. "Conditional Lemma Discovery and Recursion Induction in Hipster". *ECEASST*. 72. URL: http://journal.ub.tu-berlin.de/eceasst/article/view/1009.

Van Der Walt, P. and W. Swierstra. 2012. "Engineering proof by reflection in Agda". In: *Symposium on Implementation and Application of Functional Languages*. Springer. 157–173. DOI: 10.1007/978-3-642-41582-1_10.

"Verified cryptography for Firefox 57". URL: https://blog.mozilla.org/security/2017/09/13/verified-cryptography-firefox-57.

Voevodsky, V. 2015. "An experimental library of formalized Mathematics based on the univalent foundations". *Mathematical Structures in Computer Science*. 25(5): 1278–1294. DOI: 10.1017/S0960129514000577.

Voevodsky, V., B. Ahrens, D. Grayson, *et al.* 2011-2019. "*UniMath*: Univalent Mathematics". URL: https://github.com/UniMath.

von Neumann, J. 1993. "First Draft of a Report on the EDVAC". *IEEE Ann. Hist. Comput.* 15(4): 27–75. DOI: 10.1109/85.238389.

von Wright, J. 1994. "Program Refinement by Theorem Prover". In: *6th Refinement Workshop*. London: Springer London. 121–150. DOI: 10.1007/978-1-4471-3240-0_7.

Wadler, P. and S. Blott. 1989. "How to Make Ad-hoc Polymorphism Less Ad Hoc". In: *Proceedings of the 16th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages. POPL '89.* Austin, Texas, USA: ACM. 60–76. DOI: 10.1145/75277.75283.

Wang, M., Y. Tang, J. Wang, and J. Deng. 2017. "Premise Selection for Theorem Proving by Deep Graph Embedding". In: *Advances in Neural Information Processing Systems 30.* Curran Associates, Inc. 2786–2796. URL: http://papers.nips.cc/paper/6871-premise-selection-for-theorem-proving-by-deep-graph-embedding.pdf.

Warden, S. and F. Biancuzzi. 2009. *Masterminds of Programming: Conversations with the Creators of Major Programming Languages.* O'Reilly Media.

Watt, C. 2018. "Mechanising and verifying the WebAssembly specification". In: *Proceedings of the 7th ACM SIGPLAN International Conference on Certified Programs and Proofs.* ACM. 53–65. DOI: 10.1145/3167082.

Weirich, S., B. A. Yorgey, and T. Sheard. 2011. "Binders unbound". In: *Proceedings of the 16th ACM SIGPLAN International Conference on Functional Programming (ICFP 2011).* ACM. 333–345. DOI: 10.1145/2034773.2034818.

Wenzel, M. 2006. "Structured Induction Proofs in Isabelle/Isar". In: *Mathematical Knowledge Management.* Berlin, Heidelberg: Springer. 17–30. DOI: 10.1007/11812289_3.

Wenzel, M. 2007. "Isabelle/Isar–a generic framework for human-readable proof documents". *From Insight to Proof–Festschrift in Honour of Andrzej Trybulec.* 10(23): 277–298.

Wenzel, M. 2012. "Isabelle/jEdit – A Prover IDE within the PIDE Framework". In: *Intelligent Computer Mathematics.* Berlin, Heidelberg: Springer. 468–471. DOI: 10.1007/978-3-642-31374-5_38.

Wenzel, M. 2013a. "PIDE as front-end technology for Coq". *CoRR.* abs/1304.6626.

Wenzel, M. 2013b. "Shared-Memory Multiprocessing for Interactive Theorem Proving". In: *Interactive Theorem Proving: 4th International Conference, ITP 2013, Rennes, France, July 22-26, 2013. Proceedings.* Berlin, Heidelberg: Springer. 418–434. DOI: 10.1007/978-3-642-39634-2_30.

Wenzel, M. 2014. “Asynchronous User Interaction and Tool Integration in Isabelle/PIDE”. In: *Interactive Theorem Proving: 5th International Conference, ITP 2014, Held as Part of the Vienna Summer of Logic, VSL 2014, Vienna, Austria, July 14-17, 2014. Proceedings.* Cham: Springer International Publishing. 515–530. DOI: 10.1007/978-3-319-08970-6_33.

Wenzel, M. 2015. “Interactive Theorem Proving from the perspective of Isabelle/Isar”. In: *All about Proofs, Proofs for All.* Ed. by B. Woltzenlogel Paleo and D. Delahaye. Vol. 55. *Mathematical Logic and Foundations.* London, UK: College Publications.

Wenzel, M. 2017a. “Future Prospects of Isabelle Technology”. URL: http://sketis.net/wp-content/uploads/2017/11/Copenhagen2017.pdf.

Wenzel, M. 2017b. “Scaling Isabelle Proof Document Processing”. URL: http://sketis.net/wp-content/uploads/2017/12/Isabelle_Scaling_Dec-2017.pdf.

Wenzel, M. 2017c. “Visual Studio Code as Prover IDE for Isabelle”. URL: https://sketis.net/2017/visual-studio-code-as-prover-ide-for-isabelle.

Wenzel, M. 2018a. “Further Scaling of Isabelle Technology”. URL: https://files.sketis.net/Isabelle_Workshop_2018/Isabelle_2018_paper_1.pdf.

Wenzel, M. 2018b. “Isabelle/jEdit”. URL: http://isabelle.in.tum.de/dist/doc/jedit.pdf.

Wenzel, M. *et al.* 2004. “The Isabelle/Isar reference manual”.

Whitehead, A. N. and B. Russell. 1997. *Principia mathematica to *56.* Cambridge University Press.

Whiteside, I. J. 2013. “Refactoring proofs”. *PhD thesis.* University of Edinburgh. URL: http://hdl.handle.net/1842/7970.

Wibergh, K. 2019. “Automatic refactoring for Agda”. *MA thesis.* Chalmers University of Technology and University of Gothenburg.

Wiedijk, F. 2001. “Mizar light for HOL Light”. In: *International Conference on Theorem Proving in Higher Order Logics.* Springer. 378–393. DOI: 10.1007/3-540-44755-5_26.

Wiedijk, F. 2006. *The Seventeen Provers of the World: Foreword by Dana S. Scott (Lecture Notes in Computer Science / Lecture Notes in Artificial Intelligence)*. Berlin, Heidelberg: Springer-Verlag. DOI: 10.1007/11542384.

Wiedijk, F. 2009. “Statistics on digital libraries of mathematics”. English. *Studies in Logic, Grammar and Rhetoric*. (18(31)): 137–151.

Wiedijk, F. 2012. “Pollack-inconsistency”. *Electronic Notes in Theoretical Computer Science*. 285: 85–100. DOI: 10.1016/j.entcs.2012.06.008.

Wilcox, J. R., D. Woos, P. Panchekha, Z. Tatlock, X. Wang, M. D. Ernst, and T. Anderson. 2015. “Verdi: A Framework for Implementing and Formally Verifying Distributed Systems”. In: *Proceedings of the 36th ACM SIGPLAN Conference on Programming Language Design and Implementation. PLDI ’15*. Portland, OR, USA: ACM. 357–368. DOI: 10.1145/2737924.2737958.

Williams, T., P.-É. Dagand, and D. Rémy. 2014. “Ornaments in Practice”. In: *Proceedings of the 10th ACM SIGPLAN Workshop on Generic Programming. WGP ’14*. Gothenburg, Sweden: ACM. 15–24. DOI: 10.1145/2633628.2633631.

Wilson, S., J. Fleuriot, and A. Smaill. 2010. “Inductive Proof Automation for Coq”. In: *Second Coq Workshop*. Edinburgh, UK. URL: https://hal.inria.fr/inria-00489496.

Wimmer, S. and P. Lammich. 2018. “Verified Model Checking of Timed Automata”. In: *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*. Springer. 61–78.

Wirth, N. 1971. “Program Development by Stepwise Refinement”. *Commun. ACM*. 14(4): 221–227. DOI: 10.1145/362575.362577.

Woos, D., J. R. Wilcox, S. Anton, Z. Tatlock, M. D. Ernst, and T. Anderson. 2016. “Planning for Change in a Formal Verification of the Raft Consensus Protocol”. In: *Proceedings of the 5th ACM SIGPLAN Conference on Certified Programs and Proofs. CPP 2016*. St. Petersburg, FL, USA: ACM. 154–165. DOI: 10.1145/2854065.2854081.

Yang, K. and J. Deng. 2019. “Learning to Prove Theorems via Interacting with Proof Assistants”. In: *International Conference on Machine Learning*.

Yang, X., Y. Chen, E. Eide, and J. Regehr. 2011. "Finding and Understanding Bugs in C Compilers". In: *Proceedings of the 32nd ACM SIGPLAN Conference on Programming Language Design and Implementation. PLDI '11*. San Jose, CA, USA: ACM. 283–294. DOI: 10.1145/1993498.1993532.

Yi, K. 2006. "Educational Pearl: Proof-directed debugging revisited for a first-order version". *J. Funct. Program.* 16(6): 663–670. DOI: 10.1017/S0956796806006149.

Yu, D. and Z. Shao. 2004. "Verification of safety properties for concurrent assembly code". In: ACM. 175–188. DOI: 10.1145/1016850.1016875.

Z3 Development Team. 2008-2019. "Z3". URL: http://github.com/Z3Prover/z3/wiki.

Zalta, E. N. 2018a. "Frege's Theorem and Foundations for Arithmetic". In: *The Stanford Encyclopedia of Philosophy*. Ed. by E. N. Zalta. Fall 2018. Metaphysics Research Lab, Stanford University. URL: https://plato.stanford.edu/archives/fall2018/entries/frege-theorem/.

Zalta, E. N. 2018b. "Gottlob Frege". In: *The Stanford Encyclopedia of Philosophy*. Ed. by E. N. Zalta. Spring 2018. Metaphysics Research Lab, Stanford University. URL: https://plato.stanford.edu/archives/spr2018/entries/frege/.

Zeller, P., A. Bieniusa, and A. Poetzsch-Heffter. 2014. "Formal Specification and Verification of CRDTs". In: *Formal Techniques for Distributed Objects, Components, and Systems*. Berlin, Heidelberg: Springer. 33–48. DOI: 10.1007/978-3-662-43613-4_3.

Zhan, B. 2016. "AUTO2, a saturation-based heuristic prover for higher-order logic". *CoRR*. abs/1605.07577.

Zhang, H., G. Klein, M. Staples, J. Andronick, L. Zhu, and R. Kolanski. 2012. "Simulation Modeling of A Large Scale Formal Verification Process". In: *International Conference on Software and Systems Process*. Zurich, Switzerland: IEEE. 3–12. DOI: 10.1109/ICSSP.2012.6225979.

Zhao, J., S. Nagarakatte, M. M. K. Martin, and S. Zdancewic. 2012. “Formalizing the LLVM intermediate representation for verified program transformations”. In: *Proceedings of the 39th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2012, Philadelphia, Pennsylvania, USA, January 22-28, 2012*. 427–440. DOI: 10.1145/2103656.2103709.

Ziliani, B., D. Dreyer, N. R. Krishnaswami, A. Nanevski, and V. Vafeiadis. 2015. “Mtac: A monad for typed tactic programming in Coq”. *Journal of Functional Programming*. 25. DOI: 10.1017/S0956796815000118.

Zimmermann, T. and H. Herbelin. 2015. “Automatic and Transparent Transfer of Theorems along Isomorphisms in the Coq Proof Assistant”. *CoRR*. abs/1505.05028.

Zucker, J. 1994. “Formalization of classical mathematics in Automath”. In: *Studies in Logic and the Foundations of Mathematics*. Vol. 133. Elsevier. 127–139. DOI: 10.1016/S0049-237X(08)70202-7.