Tributes 48

60 Jahre DVMLG

DVMLG

Editors
Benedikt Löwe
Deniz Sarikaya

What can formal systems do for mathematics? A discussion through the lens of proof assistants

Angeliki Koutsoukou-Argyraki*

Department of Computer Science and Technology (Computer Laboratory), University of Cambridge, J. J. Thomson Avenue, Cambridge CB3 0FD, England

E-mail: `ak2110@cam.ac.uk`

Abstract

This article containing interviews with Jeremy Avigad, Jasmin Blanchette, Frédéric Blanqui, Kevin Buzzard, Johan Commelin, Manuel Eberl, Timothy Gowers, Peter Koepke, Assia Mahboubi, Ursula Martin, and Lawrence C. Paulson attempts to approach the question of the significance of proof assistants—in tandem with the (possible) effects of their underlying logical formalisms—for contemporary and future mathematical practice. The answer to this broad question within such a fast-developing area involving cutting-edge research cannot be clear nor complete; here, through a discussion with eleven leading experts and considering some recent advances as well as several newly started research projects, we merely attempt to illuminate varying aspects of the topic, with the hope to demonstrate its dynamic, richness and potential.

1 Introduction

1.1 Formal systems and mathematics

The typical mathematician doing research in pure or applied mathematics is hardly ever concerned with foundations. Areas of mathematical logic, such as proof theory, homotopy type theory, model theory, set theory, computability theory/recursion theory, reverse mathematics, are often considered to be closer to, or classified under computer science instead of mathematics, even though they do have interesting applications in mainstream mathematics as well. E.g., in Kohlenbach's proof mining [51, 52, 53], a research program in applied proof theory, if certain prerequisites referring to the logical form of the statement proved and the logical framework at hand are fulfilled, certain logical metatheorems guarantee the pen-and-paper extractability of effective information even from nonconstructive proofs. Proof

*The author would like to warmly thank all participants for their valuable insights and comments and Benedikt Löwe for the paper invitation. She was supported by the ERC Advanced Grant ALEXANDRIA (Project GA 742178) led by Lawrence C. Paulson.

mining has found many applications over the last two decades mainly within nonlinear analysis (e.g. there are many results in Banach spaces, fixed point theory, convex optimisation, ergodic theory, topological dynamics and more).

All along the foundational crisis in mathematics fuelled by the paradoxes of the early 20th century, the efforts by Whitehead and Russell to axiomatise mathematics and express it in symbolic logic [80], the rise and fall (due to Gödel's Incompleteness Theorems) of Hilbert's Program and all the way until today, the typical mathematician working in subfields of e.g. algebra, geometry, topology, combinatorics, number theory or analysis (even more so a mathematician doing applied mathematics) would (almost) safely ignore foundations; even constructivism and intuitionism remain to most mathematicians a foreign land.

The proof that the continuum hypothesis is independent of ZFC by the combined works of Gödel and Cohen in 1963 was followed by the proofs of the independence of a number of statements mainly within the realm of logic and set theory. In the 1992 novel "Uncle Petros and Goldbach's Conjecture" by Doxiadis [28], the main character, a reclusive former mathematician who spent his youth, talent and potential in an obsessive, fruitless attempt to prove Goldbach's conjecture, desperately tries to find consolation (or to save himself from insanity) in the belief that his inability to prove Goldbach's conjecture despite his obsessive efforts was not a failure; instead, he explains to his nephew, it was merely bad luck, as he had convinced himself that the key lies in Gödel's First Incompleteness Theorem: Goldbach's Conjecture, he now believes, happens to be a rare instance of an undecidable statement. This is of course a work of fiction, yet in reality there are indeed a few instances of statements in mathematics too that have been shown to be undecidable in ZFC (in the sense of independent of ZFC). E.g., Shelah showed in 1974 that the Whitehead Problem, a problem in group theory, is independent of ZFC [76]. Another problem independent of ZFC is Wetzel's problem, a problem on the cardinality of a set of analytic functions fulfilling certain conditions: as shown by Erdős, it depends on the truth of the continuum hypothesis [1, pp. 132–134].[1] Another more recent example from 2011 is the proof by Farah [38] and Phillips and Weaver [71] that the existence of outer automorphisms of the Calkin algebra depends on set theoretic assumptions beyond ZFC: in particular, there exist outer automorphisms assuming the continuum hypothesis, while assuming Todorčević's Axiom all automorphisms are inner.

On a different note, within computability theory, undecidable problems in the sense of not effectively solvable, that is, computational problems for which there cannot exist a computer program that always gives the

[1]The proof was very recently formalised in Isabelle/HOL by Paulson [70].

correct "yes" or "no" answer, such as Hilbert's *Entscheidungsproblem* or the Halting Problem for Turing machines, are uncountably many, yet, once again, most such known statements are within logic rather than within mainstream mathematics. The group isomorphism problem is a notable example in combinatorial group theory, with more known examples in group theory, topology, linear algebra and analysis.

Such "anomalies" remain in the sidelines and it is generally accepted that foundations are not usually considered in mainstream mathematics research; this is normally inconsequential.

But nowadays foundations play another role, too: they are blueprints for proof assistants (also known as interactive theorem provers), which brings us to the main topic of this paper.

1.2 Proof assistants and mathematics

There are several reasons why formalising mathematics with a proof assistant can be useful. I have elaborated some reasons in an article [55]; I summarise these below. I would also like to point to a new preprint by Buzzard summarising his upcoming invited talk at the *2022 International Congress of Mathematicians* [17] as well as an earlier opinion piece by the same author for the *Notices of the London Mathematical Society* [14].

Obviously a first reason is verification. This however mainly applies to formalisation of research results, since otherwise the material has usually been checked by a great number of people over the years. A second reason is that contributing to the libraries of formal proofs amounts to the creation of a database with a huge potential. Formalised material could be used in the future to create new tools with the help of artificial intelligence to discover new mathematical results. A vision for the future is the creation of an interactive assistant that would provide "brainstorming" tips to research mathematicians in real time assisting them in the process of discovering (or inventing) a new result. In any case, a library written in code offers many possibilities: it is something we can modify, interact with, reuse, in contrast with a "physical" library consisting of printed books. A third reason is that the process of formalising in itself can help the user gain brand new insights even in already familiar topics. This is not only because a formalised proof must be written in a very high level of detail, but also because using new tools forces to look at familiar material from a new perspective. Last but not least, formalising mathematics with a proof assistant can also serve educational purposes.

An important milestone that indicates that formalised mathematics and proof assistants are gradually becoming integrated into mathematical practice is that the new 2020 Mathematics Subject Classification [3] includes for the first time [29] a new class (68Vxx) referring to topics such as computer assisted proofs, proofs employing automated/ interactive theorem provers, formalisation of mathematics in connection with theorem provers.

Some interesting very recent developments in the area of formalisation of mathematics with proof assistants are listed below. Summarising the most notable developments of the last couple of years in such a rapidly developing field would be no easy task; this is a collection of some highlights that I am aware of rather than a comprehensive review—I should apologise in advance for any omissions. The interested reader is also invited to explore e.g. the Archive of Formal Proofs and mathlib, the main databases for Isabelle and Lean respectively, that are growing at a very fast pace, day by day. Monthly progress in mathlib is summarised in the Lean Community Blog. Other extensive libraries are e.g. these of Coq and Agda.

A very important recent development is the *Liquid Tensor Experiment*: it involves the formalisation (in Lean) of material from the area of Condensed Mathematics. This is a theory that Clausen and Scholze started to develop almost four years ago: it claims that topological spaces are the wrong definition, and that they should be replaced with the different notion of condensed sets. Condensed abelian groups constitute a variant of topological abelian groups, but with more convenient properties. In late 2020, Scholze posed a challenge [74] which was realised by the Lean community very soon: in subsequent blogposts ([75] and more recently [19]) progress on the project has been reported. This achievement has gained extensive publicity and has been covered by *Nature* [18] and *Quanta* [48].

This is not the first important work by the Fields medalist Peter Scholze that was formalised in Lean: Perfectoid spaces, a special kind of adic spaces of fundamental significance introduced by Scholze in 2012 [73] have recently been formalised by Buzzard, Commelin and Massot [15].

There has been considerable recent activity in the area of number theory: *Lean Forward: Usable Computer-Checked Proofs and Computations for Number Theorists* led by Jasmin Blanchette is a major ongoing research project funded by the Dutch Research Council (NWO) that got launched in 2019 and will continue until 2023. The goal of Lean Forward is to collaborate with number theorists to formalise research-level theorems and to address the main usability issues that mathematicians are confronted with in their efforts to adopt proof assistants in their work. One of the major results of the project is the formalisation in Lean of Dedekind domains and class groups of global fields by Baanen, Dahmen, Narayanan, and Nuccio Mortarino Majno di Capriglio [5]. Another very notable development is the formalisation of the solution to the cap set problem by Dahmen, Hölzl, and Lewis [20] in Lean: this is a result by Ellenberg and Gijswijt published in the Annals of Mathematics in 2017 [35] addressing the size of subsets of fields that contain no 3-term arithmetic progressions.

Other recent, important progress in number theory involves the formalisation of a substantial amount of material in analytic number theory in Isabelle/HOL by Eberl [31].

Han and van Doorn have formalised in Lean the independence of the Continuum Hypothesis [45]. More recently, Gunther, Pagano, Sánchez Terraf, and Steinberg completed a formalisation of the above result in Isabelle/ZF [42].

Edmonds, Paulson, and the present author have recently formalised Szemerédi's Regularity Lemma, a major result in extremal graph theory [32]. We employed this to formalise the proofs of the Triangle Counting Lemma and the Triangle Removal Lemma and finally prove Roth's Theorem on Arithmetic Progressions, a major result in additive combinatorics on the existence of 3-term arithmetic progressions in subsets of natural numbers [33, 34]. Independently, and around the same time, Dillies and Mehta formalised the aforementioned results in Lean following a different approach [26]; their formalisations will be incorporated into mathlib in the near future.

An upcoming Special Issue on Interactive Theorem Proving in Mathematics Research of the journal *Experimental Mathematics* contains a number of papers on new contributions. Among these are a paper by Džamonja, Paulson and the present author [30] discussing formalisations [68, 69] of a number of research results in infinitary combinatorics and set theory (more specifically in ordinal partition relations, a field that deals with generalizations of Ramsey's theorem to transfinite ordinals) by Erdős and Milner [36], Nash-Williams [63], Specker and Larson [60], leading to Larson's proof of an unpublished result by Milner asserting that for all $m \in \mathbb{N}$, $\omega^\omega \to (\omega^\omega, m)$. This is of interest not only because it involves formalisation of material within an area never formalised before (to our knowledge), but also because it is a demonstration of working with Zermelo-Fraenkel set theory in higher-order logic [67], as all the formalisations were done in Isabelle/HOL. Another paper in the issue, by Li, Paulson, and the present author [59], explores the formalisation of relatively modern mainstream research papers in number theory and analysis discussing our formalisations [56, 57, 58] in Isabelle/HOL of certain irrationality and transcendence criteria for infinite series from three different research papers (by Erdős and Straus [37], Hančl [46], and Hančl and Rucki [47]). A very important work on a different topic, also included in the aforementioned special issue, discusses the formalisation in Lean of Grothendieck's schemes in algebraic geometry by Buzzard, Hughes, Lau, Livingston, Fernández Mir, and Morrison [16]. A short time later, schemes got formalised in Isabelle/HOL, too, by Bordg, Paulson, and Li [12, 13]: to make up for the lack of dependent type theory in Isabelle/HOL, locales were employed instead to achieve the required level of expressiveness.

1.3 Proof assistants and formal systems: back to the main question

Like a driver who ignores the details of how their car engine works but still drives safely to their destination, the typical mathematician proof assistant user may not always be concerned with all the ramifications of the formal system their proof assistant of choice is based on. This is typically not true of proof assistant developers. In any case, the issue of foundations, usually hidden in the background, becomes inevitably more relevant when using a proof assistant compared to when doing mathematics with pen and paper.

Today there is a number of different proof assistants, based on different formal systems, that provide important libraries of formalised mathematical proofs. We can distinguish three big families: the proof assistants based on set theory (e.g. Mizar, Metamath); these based on simple type theory (e.g. HOL4, HOL Light, Isabelle); and these based on dependent type theory (e.g. Coq, Agda, Lean, PVS). The interested user who would like to see a direct comparison between their corresponding languages through examples of formalised material is referred to *Formalizing 100 Theorems*, a list of central mathematical theorems formalised in different proof assistants, maintained by Freek Wiedijk on his website. (Another source for comparisons, [81], though very useful, being one and a half decades old it is dated, as Lean [62], a very widely used proof assistant especially among mathematicians, would enter the picture seven years later, in 2013).

This pluralism invites us to explore to what extent logical foundations might have an effect here, i.e., within which aspects of proof assistants they manifest themselves (if at all). Thus, inevitably we are returning to the original question of the title: the question of *how formal systems themselves matter in practice for proof assistants and what they can do for mathematics*, possibly transforming the contemporary and future mathematical landscape via proof assistants.

Such a transformation may refer to the kind of research work we do; most likely it will relate to various different aspects of mathematical practice, that is, not only (or not necessarily) *what* we do but also *how* we do it. It may, e.g., refer to an evolution of the reviewing system by having the authors submit code containing formalised versions of their results along with their proofs written in "usual" mathematical language (Hales's massive Flyspeck project that succeeded in 2014 [44] to formally verify using HOL Light and Isabelle/HOL his 1998 proof of the Kepler conjecture [43] is of course well-known; we can mention a couple of examples in recent pure mathematics research, still rather isolated thus pioneering, where the formal proofs were submitted simultaneously with the original result, such as the work [50] by Kjos-Hanssen, Niraula and Yoon in metric geometry with the result formalised in Lean and the work [40] by Gouëzel and Shchur in Gromov-hyperbolic spaces with the result formalised in Isabelle/HOL). The possible upcoming transformation may moreover refer to teaching, learning and dissemination approaches in mathematics; to social aspects of large collaborative projects; or even to an increase of interest in the foundations of mathematics among mathematicians.

But let us listen to what a few of the protagonists have to say—in their own words.

2 Interviews

Jeremy Avigad, Department of Philosophy and Department of Mathematical Sciences at Carnegie Mellon University & Charles C. Hoskinson Center for Formal Mathematics.

K.-A. The new Charles C. Hoskinson Center for Formal Mathematics at Carnegie Mellon University of which you are the director, was inaugurated in September 2021. The center aims at the advancement of mathematical research by facilitating access to knowledge and resources. To this end, one of the main goals of the center is to support the development of Lean's library of formalised mathematics and of new tools to help convert mathematical statements from natural language to a formal language, as well as the creation of educational resources for the dissemination of these tools. Would you like to elaborate on how this will contribute to the evolution of mathematical practice (and the practice of related disciplines) as we know it?

Avigad. It is important to recognise that formalisation is not just about checking correctness. Building a digital mathematical library is a wonderfully collaborative activity, and the result is an important communal resource: a permanent, precise repository of mathematical knowledge, interlinked, searchable, and surveyable at any level of detail. It's like building the library of Alexandria and making it fireproof by putting it in the cloud. The resources—not just the definitions and theorems but also the algorithms and software tools that act on them—are freely available to anyone with an internet connection. They can be used to support education and discovery as much as verification.

The mission of the Hoskinson Center is to help make that technology accessible to as wide an audience as possible. Infrastructure and library development are an important part of that, but our main focus is on education and dissemination.

K.-A. Based on your rich experience with various different proof assistants (Isabelle/HOL, Coq and, more recently, Lean) what are, in a nutshell, the strengths and weaknesses of each system? What are some improvements that you would like to see in future versions of Lean?

Avigad. I don't like to make comparisons. Proof assistants are like programming languages. Everyone has their favourites, and people spend far too much time fighting over which ones are better. All three systems you mention are wonderful. All of them have yielded fundamental, groundbreaking contributions to our understanding of formal methods and what they can do.

In recent years, my focus has been on Lean. It is a beautifully designed system and it has attracted a number of energetic, enthusiastic young mathematicians and computer scientists. My favourite thing about Lean is the community that has grown around it.

In general, though, proof assistants are still too hard to use. Using the technology requires too much dedication; it's impossible to just pick it up for casual use. We need better libraries, interfaces, search engines, automation, and educational resources. Progress has been slow but steady. We'll get there.

Jasmin Blanchette, Department of Computer Science, Vrije Universiteit Amsterdam, VeriDis group at Loria, Nancy, Max-Planck-Institut für Informatik, Saarbrücken, & Institut für Systemsicherheit, Universität der Bundeswehr München.

K.-A. A benefit of using simple types instead of dependent types is the more practical implementation of efficient automation. Isabelle uses simple types and Isabelle/HOL, encoding higher-order logic, is implemented with Sledgehammer's [7, 11] automation, that calls several external automated theorem provers, giving Isabelle/HOL an advantage over other proof assistants. However, Isabelle/ZF, encoding Zermelo-Fraenkel set theory, does not feature Sledgehammer. As a leading expert in automated theorem proving and in particular Sledgehammer, would you like to elaborate on the obstacles (if any) to the implementation of Sledgehammer to Isabelle/ZF?

Blanchette. To adapt Sledgehammer to Isabelle/ZF, the three main modules of the existing Sledgehammer would have to be revisited:

(1) the relevance filter, which selects (typically) a few hundred facts (lemmas, definitions, ...) from the (typically) thousands available in background libraries;

(2) the problem translation module, which encodes Isabelle formulas into the logic of the target automatic theorem prover (ATP);

(3) the proof reconstruction module, which takes an ATP-generated proof and translates it into an Isabelle proof.

The relevance filter would be straightforward to port to ZF. It would need to be told which symbols are ZF primitives, so that it ignores them when computing the relevance of a fact with respect to the formula to prove.

The problem translation module would need the most adaptation, because it would need to translate from a different logic. I know Cezary Kaliszyk had a prototypical translation running, and the MizAR "hammer" for the Mizar proof assistant also performs a translation for another set theory, so this is possible. Most automatic theorem provers support first-order logic, and set theory is formulated in terms of that logic.

Proof reconstruction depends on having strong built-in automation directly in Isabelle/ZF that can reconstruct arbitrary ATP steps. For Isabelle/HOL, we integrated the Metis prover by Joe Hurd. One option would be to integrate Metis in Isabelle/ZF in much the same way.

Barring a lot of engineering, I see nothing really standing in the way of a Sledgehammer for Isabelle/ZF.

Frédéric Blanqui, INRIA, Laboratoire Méthodes Formelles, Université Paris-Saclay.

K.-A. You are the main proposer and Chair of the European COST Action CA20111 EuroProofNet (European Research Network on Formal Proofs) that got launched recently, in October 2021, and has over 200 participants from 30 countries. One of the main objectives of the Action is to work towards the ambitious task of achieving interoperability between different proof systems (like Coq, Isabelle/HOL, HOL Light, Agda, Lean, Matita and others). To this end, a new common logical framework (Dedukti) will be developed, along with new tools for inter-translation of proofs formalised in various different systems to and from the new common logical framework. What are some of the required characteristics of the new formal system? What do you expect will be some of the most challenging obstacles to overcome so as to achieve inter-translation between the new "global" formal system and other systems?

Blanqui. Various logical frameworks have already been developed in the past. The most prominent one, adopted by all mathematicians now, is first-order logic. In first-order logic, each mathematical theory is characterized by some propositions called axioms. E.g., Euclidean geometry and hyperbolic geometry can both be expressed in first-order logic using different axioms.

However, first-order logic suffers from a number of weaknesses. E.g., it is uneasy to express properties and proofs of objects with binding constructions, or modern categorical constructions. That is why, nowadays, many proof assistants are not based on first-order logic but rather on higher-order logic or dependent type theory, that is, a type theory where types can depend on objects.

Similarly, Dedukti is based on the simplest dependent type theory possible. But, in Dedukti, in contrast with other systems, typing is done modulo user-defined equations. Hence, while in first-order logic, theories are characterized by the symbols they use and their axioms, in Dedukti, theories are characterized by the symbols they use and their equations.

Reasoning modulo equations on types is the key feature that allows one to express in Dedukti the proofs of many systems: first-order logic and its theories, higher-order logic, as well as complex type systems like Agda and Coq.

However, we have to express in the same way the features (polymorphism, predicate subtyping, proof irrelevance, impredicativity, etc.) that are common to all systems, so that we can more easily detect the proofs that can be translated from one system to the other. E.g., Euclidean geometry and hyperbolic geometry only differ by one axiom. Hence, a proof in Euclidean geometry not using Euclid's 5th axiom can be readily translated into hyperbolic geometry.

There are many theoretical and practical challenges to scale up and fully handle all current proof assistants. A first challenge is to extend the set of proof assistant features that can be expressed in Dedukti. A first step is the article "Some axioms for mathematics" at the 2021 FSCD conference [9]. Another very important challenge is to make the translated proofs usable. This in particular requires to align the concepts used in the source system with the ones used in the target system.

Kevin Buzzard, Department of Mathematics, Imperial College London.

K.-A. Continuing a celebrated career in algebraic number theory, in recent years you have also been enthusiastically working on formal proof verification—an area that used to traditionally attract mostly computer scientists rather than mathematicians—using Lean. You are very actively involved in teaching mathematics undergraduates how to use Lean to formalise proofs, and you have been supervising many student projects in this direction, too; a new generation of mathematicians, in their formative years, is becoming accustomed to formalisation and interactive theorem proving. Watching your students learn, how do you think using a proof assistant affects the development of their mathematical thinking? Have you noticed that it may increase the students' attention to detail, or spark an interest in foundational questions? In your experience, is getting accustomed to Lean's dependent type theory actively affecting how students think about mathematics?

Buzzard. Here is what I think is going on: Students learn very quickly when coming to university that there are things which are acceptable to say in some courses, but not acceptable in others. E.g., the fact that the derivative of sine is cosine would be a perfectly acceptable claim in a 1st year mechanics course, but would probably need a detailed proof in a 1st year analysis course. I teach students how to formalise mathematics, but I am not convinced that it changes the way they solve mechanics questions one jot! I am not even convinced that it changes the way they work in their other pure mathematics courses. Perhaps it changes things slightly: maybe they are more careful with edge cases, maybe they factor out sublemmas a bit more, maybe they write more constructively. But what does this buy them in practice? Mathematicians are sometimes sloppy with edge cases—e.g., in Atiyah and MacDonald's book on commutative algebra, in their proof of the Artin-Tate lemma they do an induction where the inductive step works fine but the base case does not! An undergraduate pointed this out to me when they were formalising it in Lean and I was shocked, but Lean was right. But of course edge cases to mathematicians are just boring and trivial. Mathematicians don't need to compile their work so who cares if it is written in a modular manner. And of course, most mathematicians have no idea what constructivism is. I wonder therefore whether all that is happening is that undergraduates are learning a third "mode"—there is "applied mathematics mode", "pure mathematics mode" and then "formalisation mode". One thing I've learnt from undergraduates (showing me their problem sheets) is that pure mathematics, even at undergraduate level, is sometimes extremely difficult to formalise. A student might observe that a question on an algebraic topology example sheet which just comprises of a couple of pictures would be extremely hard to formalise, e.g., it might even be extremely hard to formally show that the question is well-defined; I see a picture of a torus being butchered in some way, but how do we know that the resulting object is truly independent of the details of the butchering? However, the students know what the lecturer is looking for in a solution, because they have seen how the lecturer does analogous questions and they understand that the point is that they are to copy the techniques and this will be an acceptable argument in this context. In particular, the student knows not to be in "formalising mode" when solving some of the problems, they just switch back into "maths how it is done in class in practice" mode.

Another observation would be that there is this inconvenience of type theory: types cannot intersect nontrivially. Sets of course can, so in ZFC, it's easy to switch between the notion of a "thing" being both a subgroup and a group at the same time. In type theory this is not so easy. Students learning Lean are learning how to do mathematics in type theory, but in practice a mathematician when working on paper just uses whatever theory is most convenient for the situation at the time; we are under no pressure to be consistent in practice.

I think that in conclusion I would say that I am teaching the students how to think carefully and pedantically, which is definitely a useful skill, but that mathematics is not only about this.

Johan Commelin, Mathematisches Institut, Albert-Ludwigs-Universität Freiburg.

K.-A. In parallel with your research in algebraic geometry and algebraic number theory, you are very actively involved in the Lean theorem prover community. You are currently leading the Liquid Tensor Experiment, having taken up a challenge posed by Peter Scholze, formalising cutting-edge research mathematics. In your experience so far, how do think Lean's dependent type theory, as an underlying logical formalism, has been helpful in formalising this area of mathematics in terms of expressiveness? E.g., overall, what have you and your collaborators found more challenging: expressing definitions in the language of Lean or actually working out proof steps? What do you think are some of the pros and cons of Lean's dependent type theory and what are some changes that you would like to see in future versions of Lean?

Commelin. The Liquid Tensor Experiment showed that within a reasonable amount of time, difficult state-of-the-art mathematics can be formally verified; providing a tangible benefit for the working mathematician.

In my experience, this is the first time that I was genuinely *assisted* by the computer in understanding a proof. Attempts to penetrate the proof using pen and paper went nowhere. But by gradually building the formalisation (with the help of a dozen other people) I gained a solid understanding of all the moving parts, and how they precisely fit together. This proof is extremely complicated, and walks a very fine line. In that sense, it comes as no surprise that Scholze had not received any feedback on this proof from within the mathematical community more than a year after publishing this proof on his website. At the same time, Scholze explains that he had some small lingering doubts about this proof, precisely because of this complexity.

The formalisation process, and all the interaction that ensued between Scholze and the formalisation team also led to a better understanding of the structure of the proof. Scholze wrote in a blogpost [75] that for him the arithmetic nature of the proof of this analytic result was clarified. Besides that, a technical ingredient was significantly simplified.

I do not know enough about type theories to give a meaningful statement about how important dependent type theory is for the Liquid Tensor Experiment. I recognize the following important factors in the success:

(1) The community. When Scholze posed his challenge (in all proof assistants), there was an enthusiastic response in the Lean community.

There were sufficiently many mathematicians with a good working-knowledge of Lean, which led to fast and fruitful collaboration.

(2) Lean's mathematical library, mathlib. This is a large and coherent library of many topics in undergraduate mathematics, which means that all these developments can be imported and used simultaneously. Both because it is developed in a single repository via a coordinated pull request system, and because the mathlib continuous integration process ensures no name collisions and no bad interactions between simplifier lemmas.

We used extensively the sections on algebra, topology, analysis, category theory, and homological algebra. Futhermore, when things did not work right, we could refactor mathlib and improve the library. Algebra (monoids and groups), analysis (semi-normed groups) and homological algebra (complexes) were refactored as part of the project. These refactors would be merged upstream in an efficient process, which was crucial for keeping the ball rolling.

The success of the Liquid Tensor Experiment is very much a success of the community around Lean and mathlib.

Manuel Eberl, Computational Logic Group, University of Innsbruck.

K.-A. As a very experienced user of Isabelle and a prolific contributor to the Isabelle Libraries and the Archive of Formal Proofs, how useful has Sledgehammer been in your everyday formalisation work?

Eberl. I do use Sledgehammer a lot, albeit less so than other users. When I do use it, it is mostly as a tool to find relevant facts to my current goal. When I think the current goal should be doable by the automation given the correct rules but I'm not quite sure what the correct rules are, I call Sledgehammer, and if it finds a proof, I then inspect the facts it uses and construct a "proper" proof from them. I tend to prefer that to the kind of proofs that Sledgehammer generates. Other people tend to use Sledgehammer regularly to find proofs and keep them that way. So I think I would not miss it quite as much as a lot of other people if it did not exist, but I certainly use it often enough that it would impede my productivity if it did not exist.

K.-A. Is the lack of Sledgehammer the reason that Isabelle/ZF is much less used (compared to Isabelle/HOL), or are there other reasons related to the expressiveness of Isabelle/ZF?

Eberl. First of all, I must say that I have never used Isabelle/ZF, so take everything I say with a huge grain of salt. My impression is that there is no problem at all with the expressiveness of Isabelle/ZF. The major reason why it is not used is probably simply that the libraries and tooling (proof automation, probably definitional tools) are nowhere near as nice as those of Isabelle/HOL. As for why that is, I suspect that a large amount of it is simply historical: Isabelle/HOL got quite popular, so many people flocked to it and built libraries and tools, and that made it even more popular etc. Nothing comparable happened for Isabelle/ZF.

From what I understand, there are technical issues with using "untyped" logics such as ZF in a theorem prover. Working with the "naked" logic is possible, but tedious. For productive work, you want something like Mizar's soft types, and overloaded operators, and management of algebraic structures like groups (perhaps using some analogue of type classes). All of that takes quite a bit of engineering. I understand that there were (or still are) people working on such things, and I'm sure they can in principle be brought into a state were they work well enough to make working in Isabelle/ZF just as pleasant as in Isabelle/HOL, but it is a lot of work, and it takes quite a lot of energy to get it to a "self-sustaining" point where it is good enough that it will attract other people to work on it.

K.-A. Do you think you can make a prognosis about the possible limitations of simple type theory?

Eberl. In the kind of mathematics that I formalise, I have not really encountered any hard obstacles due to using simple type theory. There are definitely hard limitations, such as the inability to quantify over types, which makes some definitions basically impossible to write down. And as far as I know, there are also other logical issues when it comes to formalising category theory or advanced set-theoretic arguments like forcing, but I do not know much about these fields. In my experience, such things do not crop up too often in practice. The more interesting limitations are where something is possible to do in simple type theory, but becomes significantly more painful.

The one part of the Isabelle library where such issues become most apparent to me is in abstract algebra: the obvious way to define, e.g., a group is as a type with a binary operation and a neutral element that fulfil certain laws. The type classes in Isabelle/HOL make this approach very comfortable, which is why that is the way it is done in the standard library. But as soon as we want to talk about different groups and their relations to one another and their subgroups etc., this does not work without dependent types. Instead, one can define groups with an explicit carrier set and carry the group operations and the group axioms around manually. This is the approach that the HOL-Algebra library (also in Isabelle/HOL) follows, but it is much less pleasant to use than the type-based approach from the standard library.

A similar situation exists, e.g., for topologies: there is a type-class-based version where the topology comes from a type class (where you can just write "`open` X" for "the set X is open"), and a more general one where a topology is a separate object (where you write "`openin top` X" for "the set X is open with respect to the topology `top`"). Having two separate libraries that do the same thing in different ways is always problematic for maintainability and usability. There is some tooling that allows transferring results between the two approaches, but it is unfortunately not as pleasant to use as one would want.

In any case, these problems are not show stoppers. They may force you to write down some things in a less natural fashion than you would have to in a stronger logic, but this extra effort always has to be seen in conjunction with the advantages of simple type theory: as far as I am aware, simple type theory-based systems like Isabelle/HOL tend to have better performance and better proof automation than systems using stronger logics. Depending on the concrete application, this may offset the less expressive logic.

Sir Timothy Gowers, Collège de France, Department of Pure Mathematics and Mathematical Statistics, University of Cambridge, & Trinity College, Cambridge.

K.-A. You are a strong supporter of proof assistants and the formalisation of mathematics; for many years, during your celebrated career in mathematics, you have been advocating the vision of developing an interactive tool that would provide direct assistance to working mathematicians in their everyday research by offering ideas, hints, information related to the work at hand and proving auxiliary results [39, 41, 4]. Given the state of the art of the tools and the most recent advances in the area of proof assistants and formalisation, how optimistic are you that this vision is getting closer to realisation?

Gowers. As background, I should say that my main interest in this sphere is not so much formalisation as fully automatic proof discovery, but the two are sufficiently closely related that I am certainly interested in following what other people are doing in the formalisation area.

My answer to your first question is that I am very uncertain. I think there is certainly the potential for a lot of non-revolutionary progress, which I think might be enough to develop tools of genuine use to mathematicians. Indeed, I hope to contribute to that development by continuing to work on human-oriented theorem proving, but of course there are other ways that progress may well be made, such as machine learning. (My hunch about machine learning is that there is a fundamental limitation to what it can do if it is operating entirely on its own—roughly speaking corresponding to the need to have non-obvious mathematical ideas. So I expect to see impressive progress in the short term but also to see a plateau being reached unless it is combined with more traditional approaches. But it's always dangerous to make predictions like that.)

My short answer to your question is that I am optimistic that there will be exciting progress over the next ten years or so, but I think we are probably two or three big breakthroughs away from a system that can solve interesting research problems. (But a lot depends on what one means by "solve" here, and in particular on how much human assistance there is.)

K.-A. What are some theorems or research areas that you would like to see formalised in the near future?

Gowers. I suppose I'd answer that I don't have a direct interest in any particular part of mathematics becoming formalised, because the areas I work in tend to be ones where we are not too worried about whether the main results are correct. (I contrast that with some areas where major results sometimes depend on other major results that have never been properly written down, and things like that. Such situations are quite rare in combinatorics.) So I could go in one of two ways. Either I'd say that for the health of mathematics it would be good to identify the more "troubled" areas where the foundations are potentially shaky and put those right, or I'd simply say that I'd find it personally satisfying to see the main results in my own area formalised. Or I could be more selfish still and say that in the past I have written some pretty complicated papers that almost certainly have details that aren't fully correct, so it would be a good feeling if some of them were formalised—if I ever found out that somebody was interested in doing that, I would be more than happy to cooperate in finding any necessary corrections. (I do not believe that any of my papers are incorrect in a worrying way, but some of the fixes could be quite hard to find.)

K.-A. You initiated the first Polymath project in 2009. Do you think that thanks to the rise of proof assistants massively collaborative mathematics may soon become customary?

Gowers. I think that what's going on with formalisation could be described already as a massively collaborative project, though it's not precisely the same as mathematics. The way things look at the moment, people are too wedded to the current reward structures of mathematics for the massively collaborative approach to become the main way of operating, and it can be hard to organize a successful project unless one has a strong online presence. So I think that there will continue to be Polymath projects, but I don't think they are going to become mainstream in the near future.

Peter Koepke, Mathematisches Institut, Rheinische Friedrich-Wilhelms-Universität Bonn.

K.-A. Proof assistants use formal languages that are reminiscent more of computer code rather than "natural" mathematical language and notation that is familiar to mathematicians. The goal of natural proof assistants is to remedy this issue, thus making formalised proofs more easily readable and proof assistants more user-friendly to mathematicians. An effective combination of the formal rigour of logical calculi behind proof assistants with the use of natural mathematical language and notation as in usual mathematical textbooks and research papers would be an extremely powerful development. As one of the creators of the Isabelle/Naproche [24, 25] natural proof assistant, would you like to share some insights about choosing an appropriate underlying logical formalism that can nicely combine with natural language?

Koepke. We focus on the natural language and structuring of mathematical texts as they are really employed in mathematical publications. In an appropriate context of definitions, notations, and axioms, Naproche is, e.g., able to directly accept, process and proofcheck (a LaTeX source of) the following formulation of the open mapping theorem from complex analysis, including natural language phrases and mathematical symbolisms:

Theorem 2.1 (Open Mapping Theorem)**.** Assume f is a holomorphic function and $B_\varepsilon(z)$ is a subset of the domain of f. If f is not constant on $B_\varepsilon(z)$ then $f[B_\varepsilon(z)]$ is open.

This kind of mathematics is usually modelled by first-order logic and some version of set theory. We have inherited the first-order approach from our predecessor system SAD (System for Automated Deduction, by Varchinine, Lyaletski, and Paskevich [79]). An important advantage is that the leading automated theorem provers which Naproche continuously calls for minor proof tasks use first-order logic themselves. As natural language and the language of mathematics use "soft types" it may be better in the long run to switch to an appropriate formal language between first-order logic and type theory.

K.-A. What are the reasons behind your choice of Isabelle?

Koepke. Originally we have chosen Isabelle as a proof development environment in which we can comfortably run and use the Naproche program. So far there has been no connection to the typical Isabelle logics like Isabelle/HOL. We are, however, collaborating with Makarius Wenzel to find ways to map logical entities of Naproche to Isabelle and vice versa. We have an experimental setup where Naproche is invoking the Isabelle/Sledgehammer mechanism to discharge certain proof obligations.

K.-A. How has your long career in set theory and foundations of mathematics influenced your taste in logical formalisms for proof assistants?

Koepke. My experience with first-order set theory of course came in handy for dealing with SAD/Naproche formalisations and with the logical mechanisms of the system. Similar to Naproche, the working language of set theory introduces all sorts of notions like numbers, functions, structures, etc. on top of the frugal language of the axioms. But if other formalisms accommodate actual natural language better I shall happily adopt them. The trouble with natural (mathematical) language are the freedoms and ambiguities that people use to convincingly express their intuitions and arguments. First-order logic is able to deal with all sorts of exceptions by "if ... then ... otherwise ..." constructs, whereas other formalisms might force users to write in a more regulated and possibly "unnatural" language.

Assia Mahboubi, INRIA, Gallinette Team, Nantes & Vrije Universiteit Amsterdam.

K.-A. You are a very experienced user of the Coq proof assistant and you are particularly interested in the interplay between computer algebra systems and formal proofs. Your new EU-funded FRESCO project ("Fast and Reliable Symbolic Computation"), that started in November 2021, aims to explore whether computer algebra systems could be both reliable and fast. To this end, the programming features of proof assistants will be enriched to become efficiently combined with computer algebra systems, while compatibility with logical foundations will be preserved. Undoubtedly, the combination of the computational power of computer algebra systems with the formal reasoning tools provided by proof assistants based on various logical formalisms would be a very powerful combination. Would you like to give a few examples of applications in mathematical practice and research where we could see significant advances thanks to this interplay? And would you like to comment on which logical formalism(s)/formal system(s) you believe would be more preferable or practical to combine with computer algebra systems?

Mahboubi. From experimentation to proofs, there is a tremendous momentum right now for computer-aided mathematics. The use of computers for formulating conjectures, but also for substantiating proof steps, pervades mathematics, even in its most abstract fields. Most of these computer proofs are produced by symbolic computations, using computer algebra systems. Sadly, these systems suffer from severe, intrinsic flaws, key to their amazing efficiency, but preventing any flavour of post-hoc verification.

The field of number theory is an emblematic example of this change of era, and computers have become a crucial instrument for basic research in number theory, both for designing conjectures and for proving them. The Birch and Swinnerton-Dyer's conjecture about the rational points of an abelian variety, is a notorious example of a computer-shaped hunch, based on (at the time) intensive calculations. This fertile conjecture has sparked fundamental contributions, although if the problem remains open at the time of writing: the Clay Mathematics Institute even advertises a 1 million dollar bounty for its solution. Regarding published proofs, prominent examples include Bharghava's work in the sub-field of number theory called "geometry of numbers", crowned by a Fields medal in 2014 or Helfgott's groundbreaking proof of the long-standing Ternary Goldbach conjecture, in analytic number theory.

Verified (and fast) computer algebra would of course provide a principled way to assess the correctness of such achievements, when the social process of peer-reviewing just falls short of evaluating the proofs produced by computers. But it would also provide an invaluable tool for gaining confidence in the intuitions shaped by computer calculations. These calculations can be of a very diverse nature: evaluation of formulas, simplification of algebraic expressions, plots, etc. But these can all go very wrong, and for subtle reasons, which pertain to the semantic of symbolic computations (or to the lack thereof) rather than to the bugs in the usual sense, that of programming errors. Another important application I can foresee is the validation of handbooks and databases, gathering collections of mathematical objects together with their properties, like the celebrated NIST Handbook of Special Functions or the reference L-functions and Modular Forms Database, in number theory.

Modern verification tools can be classified according to the expressivity of the logical language available to state specifications, which in turn impacts their ability to automate proof search. E.g., variants of propositional logic have earned their stripes in hardware design, but specifying and verifying cyber-physical systems requires first-order logic and beyond. Verifying computer algebra in the large is more demanding, as elementary specifications will casually involve quantifying over objects such as "finite fields of an arbitrary characteristic p", with a formal integer p. Such a parametrisation is typically beyond the skills of computer algebra systems; they only provide concrete instances of these fields, for concrete values of p, as this prime integer controls algorithmic choices in modular arithmetic. In fact, verifying computer algebra in the large calls for a first-class, representation of hierarchies of mathematical structures in the logic, a feature which is available in dependent type theory.

Ursula Martin, School of Informatics, University of Edinburgh & Wadham College, Oxford.

K.-A. Your celebrated career covers many areas in theoretical computer science and formal methods; in recent years, supported by an EPSRC Fellowship, you have also been investigating the nature of mathematical practice as a social machine with your project "The Social Machine of Mathematics". In addition to various online databases (such as arXiv, Google Scholar, ResearchGate, MathSciNet, Orcid...) having revolutionised access to information and knowledge, various online platforms such as MathOverflow, MathStackExchange, GitHub and Zulip (and sometimes, even social media) actively facilitate communication and collaboration. The latter tools have been broadly adopted by the proof assistant community and contributed to its activity and success. Do you think that mathematical practice is, in this way, being radically transformed? If yes, do you perhaps see more working mathematicians taking more interest in the foundations of mathematics due to interest in proof assistants? And do you think that the existence of many different proof assistants is affecting (either accelerating or hindering) the progress of this transformation?

Martin. It's hard to recall, even for those of us old enough to remember, what the practice of mathematics was like before the personal computer; before email; before the internet; before LaTeX; before the arXiv; before collaboration sites like MathOverflow and GitHub; and before the massive expansion of mathematical publication that these now ubiquitous tools have enabled, with mathematical publications quadrupling since 1996.[2]

Yet what has remained remarkably stable is the notion of an academic mathematical paper: a document, now digital rather than paper, but otherwise still published in a journal sponsored by a recognised entity, attributed to a small number of named authors, accredited by the journal's refereeing process, with a fairly standard kind of structure, content, argument and citation practice established within particular subdisciplines. While there are undoubted triumphs of the use of proof assistants—e.g., Gonthier's work on the Odd-Order Theorem, or Hales's proof of the Kepler conjecture—the contribution to published mathematical papers of proof assistants is at a low level, by contrast with the contribution of software such as GAP, Mathematica or MATLAB. Thus, while one might argue that academic mathematical practice has the potential to be radically transformed by proof assistants, it is less easy to claim that this has so far happened.

To look at the reasons for this, one might look, not at the nuances of particular proof assistants, but more generally at how technological, or

[2]Cf., e.g., the World Report on the Scimago Journal & Country Rank webpage, listing 53,564 mathematics publications in 1996 and 183,582 mathematics publications in 2019.

other, change comes about within the institutions involved in the creation of academic mathematics: universities, funding bodies, and publishers, and at how institutions, or individuals within them, are incentivized to bring about far-reaching changes to current practices, when such practices, e.g., journal publications, are so entrenched in traditional mechanisms for credit and reward. It is noteworthy, e.g., that Gonthier did his work while employed by a company (Microsoft Research), and one might speculate on whether Hales would have felt able to do his lengthy machine proof of the Kepler conjecture if he had not had tenure. By contrast, in commercial software development, if proof technology is needed to achieve the commercial ends of the company, the infrastructure and workflow can be restructured to accommodate it—cf., e.g., Facebook's use of the Infer static analyser, which is based on separation logic [27].

As to the impact of having many different software tools—proof assistants among them—certainly uptake and use of such tools might be enhanced if a single proof assistant became as ubiquitous and as taken for granted as LaTeX, Maple or Jupyter Notebooks, or indeed embedded in such platforms. Is the reason for a diversity of tools a positive one—the need the research community feels, at this stage of development, for diversity and for experiment rather than being forced into a common approach? Or is it rather a lack of institutional or individual incentives, enthusiasm, leadership or skill for the trade-offs and political activity involved in uniting behind a common approach?

Lawrence C. Paulson, Computer Laboratory, University of Cambridge & Clare College, Cambridge.

K.-A. You are the Principal Investigator of the ERC Project "ALEXANDRIA—Large Scale Formal Proof For the Working Mathematician" which started in Autumn 2017 at the University of Cambridge and aims at the investigation of the formalisation of mathematics in practice using Isabelle/HOL and, more broadly, at contributing to the creation of a proof development environment attractive to working mathematicians. What do you think are the main takeaways from the project so far?

Paulson. ALEXANDRIA has made it possible to engage with the mathematical community and fully explore the issues surrounding the formalisation of mathematics. This engagement was primarily but not exclusively with Isabelle/HOL, since some experiments were also done using Lean. Roughly speaking, our objectives were to explore what could be accomplished using our tools, and where weaknesses were identified, to try to mitigate them.

I've been immensely satisfied with our progress. A group of people from quite different backgrounds have pursued a variety of subprojects, some individually and some in smaller groups, sometimes with outside collaborators. We accomplished things that some observers thought were impossible, notably the formalisation of schemes in Isabelle/HOL [12, 13]. Time and again we saw that advanced material such as Szemerédi's Regularity Lemma [32] weren't really that difficult and much of the difficulty we did encounter lay in the errors and inconsistencies of our source material.

A separate but essential effort was aimed at reducing the labour that goes into formalisation. This produced the SErAPIS search engine [77, 78], which is the first example of true intelligent search for an interactive theorem prover: material from all the libraries can be sought on the basis of concepts as well as syntax. Other research, less well advanced but deeply important, is aimed at using machine learning and other technologies to recognise proof patterns and assist the interactive process of formalisation by suggesting solution fragments.

K.-A. As one of the creators [65, 64] (and a very experienced user) of Isabelle, you have stressed many times that, when it comes to different formal systems, pluralism is both important and inevitable. At the same time, the answer to the question of whether simple [66] or dependent type theory is more practical for formalising mathematics (or rather: each area of mathematics) and what is the best way to use each formalism, is far from being clear. Would you like to elaborate a bit on this topic?

Paulson. I'm glad you say that "the question of whether simple or dependent type theory is more practical for formalising mathematics... is far from being clear". Plenty of people have made up their minds already. But that's premature in such a rapidly developing field. We have seen that we can go a long way in simple type theory (interestingly, with rather little reliance on axiomatic type classes). We have not encountered any no-go areas so far. We have the benefit of extensional functions and ordinary equality as well as strong automation, and full set theory when we need it. Dependent type theories are lacking in those particular areas, but have the benefit of greater expressiveness; and as they continue to evolve, who can say what things will look like in five or ten years?

3 A final comment

Our discussion has merely scratched the surface of this vast topic, yet it is already obvious that foundations, as the basis of proof assistants, can potentially be of substantial service to mathematical practice in many different ways.

But let us conclude with acknowledging that when it comes to computers and the future of mathematics, proof assistants and foundations are only one side of the story.

This is not only because computer algebra systems, natural language processing and automatic proof discovery can provide indispensable tools too, as already mentioned in the above discussions. This is because it appears that the axiomatic/ symbolic approach as conceived by Leibniz, following his vision of *calculus ratiocinator*, could take us only so far—even assuming we eventually manage to formalise "all" mathematics as wished in the QED Manifesto from the 1990es [2]. Progress seems to require the combination of alternative approaches. An interesting analogy due to Georg Gottlob is seeing "rule knowledge and logical reasoning versus machine learning, e.g., neural networks" as "left part of the brain versus right part of the brain": they have different, but complementary functions, the former inducing rationality and the latter inducing imagination and creativity.

To achieve the creation of interactive tools that would actively and efficiently help research mathematicians in their creative work, it is also new advances in artificial intelligence and machine learning that can promise novel developments in mathematical practice through their applications to automated theorem proving and proof assistants. E.g., pattern recognition tools from machine learning could find applications not only in searching the libraries of formal proofs, but also in recognising proof patterns and providing proof recommendation methods thus enhancing automation. Some promising efforts in this direction are, e.g., [8, 54, 61, 6, 72]. A new book by Holden [49] provides a comprehensive review of research applications on incorporating machine learning into automated theorem proving which has the potential of developing tools that could improve automation tools featured by interactive theorem provers.

The communities of machine learning and formal verification have indeed been growing increasingly close during the past few years. Some highlights are successful ongoing conference series such as *Artificial Intelligence and Theorem Proving* (AITP) and *Intelligent Computer Mathematics* (CICM) as well as workshops such as MATH-AI "The Role of Mathematical Reasoning in General Artificial Intelligence" organised within ICLR 2021. The ERC funded projects AI4REASON (2015–2020) led by Josef Urban at the Czech Technical University in Prague (CTU) and SMART (2017–2022) led by Cezary Kaliszyk at the University of Innsbruck have in particular produced a vast number of results. It is also worth noting that one of the six working groups within the new European COST Action CA20111 EuroProofNet (mentioned previously) is devoted to *Machine learning in proofs* (WG5) and led by Kaliszyk.

Though unrelated to proof assistants, it is worthwhile to mention an impressive breakthrough in machine learning for pure mathematics that has been very recently achieved thanks to a collaboration between DeepMind researchers and research mathematicians [21, 22]. More specifically, important results in knot theory [23] and representation theory [10] have been achieved by employing machine learning tools. This reinforces the hope that artificial intelligence in itself is powerful enough to provide useful auxiliary technology for research mathematics—and thus may be a very promising candidate for progress acceleration when combined with proof assistants too.

Bibliography

[1] Aigner, M. & Ziegler, M., *Proofs from The Book*, (5th ed.), Springer-Verlag, Berlin, 2014.

[2] Anonymous, The QED Manifesto. In: Bundy, A. (ed.), *Automated Deduction. CADE-12. 12th International Conference on Automated Deduction, Nancy, France, June 26–July 1, 1994, Proceedings.* Lecture Notes in Computer Science, Vol. 814, Springer, 1994, pp. 238–251.

[3] Associate Editors of Mathematical Reviews and zbMATH, MSC2020. Mathematics Subject Classification System, 2020.

[4] Ayers, E. W., Gowers, W. T., & Jamnik, M., A Human-Oriented Term Rewriting System. In: Benzmüller C., Stuckenschmidt H. (eds.), *KI 2019: Advances in Artificial Intelligence. 42nd German Conference on AI, Kassel, Germany, September 23–26, 2019, Proceedings.* Lecture Notes in Computer Science, Vol. 11793. Springer, Cham, 2019, pp. 76–86.

[5] Baanen, A., Dahmen, S. R., Narayanan, A., & Nuccio Mortarino Majno di Capriglio, F. A. E., A Formalization of Dedekind Domains and Class Groups of Global Fields. In: Cohen, L. and Kaliszyk, C. (eds.), *12th International Conference on Interactive Theorem Proving, ITP 2021, June 29 to July 1, 2021, Rome, Italy (Virtual Conference).* Leibniz International Proceedings in Informatics (LIPIcs), Vol. 193. Leibniz-Zentrum für Informatik, 2021, pp. 5:1–5:19.

[6] Bansal, K., Loos, S., Rabe, M., Szegedy, C., & Wilcox, S., HOList: An Environment for Machine Learning of Higher Order Logic Theorem Proving. In: Chaudhuri, K. & Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9–15 June 2019, Long Beach, California, USA*. Proceedings of Machine Learning Research, Vol. 97, MLResearchPress, 2019, pp. 454–463.

[7] Blanchette, J. C., Kaliszyk, C., Paulson, L. C., & Urban, C., Hammering towards QED. Journal of Formalized Reasoning, 9:1, 101–148, 2016.

[8] Blanchette, J. C., Greenaway, D., Kaliszyk, C., Kühlwein, D., & Urban, J., A Learning-Based Fact Selector for Isabelle/HOL. Journal of Automated Reasoning 57, 219–244, 2016.

[9] Blanqui, F., Dowek, G., Grienenberger, É, Hondet, G., & Thiré, F., Some Axioms for Mathematics. In: Kobayashi, N. (ed.), *6th International Conference on Formal Structures for Computation and Deduction, FSCD 2021, July 17-24, 2021, Buenos Aires, Argentina (Virtual Conference).* Leibniz International Proceedings in Informatics (LIPIcs), Vol. 195. Leibniz-Zentrum für Informatik, 2021, pp. 20:1–20:19.

[10] Blundell, C., Buesing, L., Davies, A., Veličković, P., & Williamson, G., Towards combinatorial invariance for Kazhdan-Lusztig polynomials. Preprint, 2021 (arXiv:2111.15161).

[11] Böhme, S. & Nipkow, T., Sledgehammer: Judgement Day. In: Giesl, J. & Hähnle, R. (eds.), *Automated Reasoning, 5th International Joint Conference, IJCAR 2010, Edinburgh, UK, July 16–19, 2010. Proceedings.* Lecture Notes in Computer Science Vol. 6173. Springer, 2010, pp. 107–121.

[12] Bordg, A., Paulson, L. C., & Li, W., Simple Type Theory is not too Simple: Grothendieck's Schemes without Dependent Types. Preprint, 2021 (arXiv:2104.09366).

[13] Bordg, A., Paulson, L.C., & Li, W., Grothendieck's Schemes in Algebraic Geometry. Archive of Formal Proofs, 2021, 29 March 2021.

[14] Buzzard, K., Computers and Mathematics. Newsletter of the London Mathematical Society, 484, 32–36, 2019.

[15] Buzzard, K., Commelin, J., & Massot, P., Formalising perfectoid spaces. In: Blanchette, J. & Hritcu, C. (eds.), *Proceedings of the 9th ACM SIGPLAN International Conference on Certified Programs and Proofs, CPP 2020, New Orleans, LA, USA, January 20–21, 2020.* Association for Computing Machinery, 2020, pp. 299–312.

[16] Buzzard, K., Hughes, C., Lau, K., Livingston, A., Fernández Mir, R., & Morrison, S., Schemes in Lean. To appear in: Experimental Mathematics (doi: 10.1080/10586458.2021.1983489).

[17] Buzzard, K., What is the point of computers? A question for pure mathematicians. Preprint, 2022 (arXiv:2112.11598v2).

[18] Castelvecchi, D., Mathematicians welcome computer-assisted proof in 'grand unification' theory. Nature 595, 18–19, 2021.

[19] Commelin, J., Liquid Tensor Experiment: an update. Blog post on the Lean community blog, 31 December 2021.

[20] Dahmen, S. R., Hölzl, J., & Lewis, R. Y., Formalizing the solution to the cap set problem. In: Harrison, J., O'Leary, J., & Tolmach, A. (eds.), *10th International Conference on Interactive Theorem Proving, ITP 2019, September 9–12, 2019, Portland, OR, USA*. Leibniz International Proceedings in Informatics (LIPIcs), Vol. 141. Leibniz-Zentrum für Informatik, 2019, pp. 15:1–15:19.

[21] Davies, A., Veličković, P., Buesing, L., Blackwell, S., Zheng, D., Tomašev, N., Tanburn, R., Battaglia, P., Blundell, C., Juhász, A., Lackenby, M., Williamson, G., Hassabis, D., & Kohli, P., Advancing mathematics by guiding human intuition with AI. Nature 600, 70–74, 2021.

[22] Davies, A., Kohli, P., & Hassabis, D., Exploring the beauty of pure mathematics in novel ways. Blog post on the DeepMind blog, 1 December 2021.

[23] Davies, A., Juhász, A., Lackenby, M., & Tomasev, N., The signature and cusp geometry of hyperbolic knots. Preprint, 2022 (arXiv:2111.15323v2).

[24] De Lon, A., Koepke, P., Lorenzen, A., Marti, A., Schütz, M., & Wenzel, M.. The Isabelle/Naproche Natural Language Proof Assistant. In: Platzer, A. & Sutcliffe, G. (eds.), *Automated Deduction. CADE 28. 28th International Conference on Automated Deduction, Virtual Event, July 12–15, 2021, Proceedings.* Lecture Notes in Computer Science, Vol. 12699, Springer, 2021, pp. 614–624.

[25] De Lon, A., Koepke, P., Lorenzen, A., Marti, A., Schütz, M., & Sturzenhecker, E., Beautiful Formalizations in Isabelle/Naproche. In: Kamareddine, F. & Sacerdoti Coen, C. (eds.), *Intelligent Computer Mathematics. 14th International Conference, CICM 2021, Timisoara, Romania, July 26–31, 2021, Proceedings.* Lecture Notes in Computer Science, Vol. 12833, Springer, 2021, pp. 19–31.

[26] Dillies, Y. & Mehta, B., Formalising Szemerédi's Regularity Lemma in Lean. In: Andronick, J. & de Moura, L. (eds.), *13th International Conference on Interactive Theorem Proving (ITP 2022).* Leibniz International Proceedings in Informatics, Vol. 237, Schloss Dagstuhl, 2022, pp. 9:1–9:19.

[27] Distefano, D., Fähndrich, M., Logozzo, F., & O' Hearn, P. W., Scaling static analyses at Facebook. Communications of the ACM 62, 62–70, 2019.

[28] Doxiadis, A., *Uncle Petros and Goldbach's Conjecture.* Bloomsbury, 2000.

[29] Dunne, E. & Hulek, K., Mathematics Subject Classification 2020. EMS Newsletter, March 2020.

[30] Džamonja, M., Koutsoukou-Argyraki, A., & Paulson, L. C., Formalising Ordinal Partition Relations Using Isabelle/HOL. To appear in: Experimental Mathematics (doi: 10.1080/10586458.2021.1980464).

[31] Eberl, M., Nine Chapters of Analytic Number Theory in Isabelle/HOL. In: Harrison, J., O'Leary, J., & Tolmach, A. (eds.), *10th International Conference on Interactive Theorem Proving, ITP 2019, September 9–12, 2019, Portland, OR, USA.* Leibniz International Proceedings in Informatics (LIPIcs), Vol. 141. Leibniz-Zentrum für Informatik, 2019, pp. 6:1–6:19.

[32] Edmonds, C., Koutsoukou-Argyraki, A., & Paulson, L. C., Szemerédi's Regularity Lemma. Archive of Formal Proofs, 2021, 5 November 2021.

[33] Edmonds, C., Koutsoukou-Argyraki, A., & Paulson, L. C., Roth's Theorem on Arithmetic Progressions. Archive of Formal Proofs, 2021, 28 December 2021.

[34] Edmonds, C., Koutsoukou-Argyraki, A., & Paulson, L. C., Formalising Szemerédi's Regularity Lemma and Roth's Theorem on Arithmetic Progressions in Isabelle/HOL. Preprint, 2022 (arXiv:2207.07499).

[35] Ellenberg, J. S. & Gijswijt, D., On large subsets of $\mathbb{F}_q^n$ with no three-term arithmetic progression. Annals of Mathematics 185:1, 339–343, 2017.

[36] Erdős, P. & Milner, E. C., A theorem in the partition calculus. Canadian Mathematical Bulletin 15:4, 501–505, 1972.

[37] Erdős, P. & Straus, E. G., On the irrationality of certain series. Pacific Journal of Mathematics, 55:1, 85–92, 1974.

[38] Farah, I., All automorphisms of the Calkin algebra are inner. Annals of Mathematics, 173:2, 619–661, 2011.

[39] Ganesalingam, M. & Gowers, W. T., A fully automatic theorem prover with human-style output. Journal of Automated Reasoning 58:2, 253–291, 2017.

[40] Gouëzel, S. & Shchur, V., A corrected quantitative version of the Morse lemma. Journal of Functional Analysis, 277:4, 1258–1268, 2019.

[41] Gowers, W.T., Rough Structure and Classification. In: Alon N., Bourgain J., Connes A., Gromov M., & Milman V. (eds.), *Visions in Mathematics. GAFA 2000 Special Volume, Part I.* Modern Birkhäuser Classics. Birkhäuser, Basel, 2010, pp. 79–117.

[42] Gunther, E., Pagano, M., Sánchez Terraf, P., & Steinberg, M., The Independence of the Continuum Hypothesis in Isabelle/ZF. Archive of Formal Proofs, 2022, 6 March 2022.

[43] Hales, T. C., A proof of the Kepler conjecture. Annals of Mathematics, 162:3, 1065–1185, 2005.

[44] Hales, T., Adams, M., Bauer, G., Dang, T. D., Harrison, J., Hoang, L. T., Kaliszyk, C., Magron, V., McLaughlin, S., Nguyen, T. T., Nguyen, Q. T., Nipkow, T., Obua, S., Pleso, J., Rute, J., Solovyev, A., Ta, T. H. A., Tran, N. T., Trieu, T. D., Urban, J., Vu, K., & Zumkeller, R., A Formal Proof of the Kepler Conjecture. Forum of Mathematics, Pi, 5, e2, 2017.

[45] Han, J. M. & van Doorn, F., A Formal Proof of the Independence of the Continuum Hypothesis. Preprint, 2021 (arXiv:2102.02901).

[46] Hančl, J., Irrational rapidly convergent series. Rendiconti del Seminario Matematico della Università di Padova, 107, 225–231, 2002.

[47] Hančl, J. & Rucki, P., The transcendence of certain infinite series. Rocky Mountain Journal of Mathematics, 35:2, 531–537, 2005.

[48] Hartnett, K., Proof Assistant Makes Jump to Big-League Math. Quanta Magazine, 28 July 2021.

[49] Holden, S. B., Machine Learning for Automated Theorem Proving: Learning to Solve SAT and QSAT. Foundations and Trends in Machine Learning, 14:6, 807–989, 2021.

[50] Kjos-Hanssen, B., Niraula, S., & Yoon, S., A parametrized family of Tversky metrics connecting the Jaccard distance to an analogue of the normalized information distance. In: Artemov, S.N. & Nerode, A. (eds.), *Logical Foundations of Computer Science. International Symposium, LFCS 2022, Deerfield Beach, FL, USA, January 10–13, 2022, Proceedings.* Lecture Notes in Computer Science, Vol. 13137, Springer, 2022, pp. 112–124.

[51] Kohlenbach, U., *Applied Proof Theory. Proof Interpretations and their use in Mathematics.* Springer Monographs in Mathematics, Springer, 2008.

[52] Kohlenbach, U., Recent progress in proof mining in nonlinear analysis, IFCoLog Journal of Logics and its Applications, 10:4, 3357–3406, 2017.

[53] Kohlenbach, U., Proof-theoretic Methods in Nonlinear Analysis, In: Sirakov, B., Ney de Souza, P., & Viana, M. (eds.), *Proceedings of the International Congress of Mathematicians, Rio de Janeiro 2018. Volume II, Invited Lectures*, World Scientific, 2018, pp. 61–82.

[54] Komendantskaya, E., Heras, J., & Grov, G., Machine Learning in Proof General: Interfacing Interfaces. In: Kaliszyk, C. & Lüth, C. (eds.), *Proceedings 10th International Workshop On User Interfaces for Theorem Provers Bremen, Germany, July 11th 2012*, Electronic Proceedings in Theoretical Computer Science, Vol. 118, Open Publishing Association, 2013, pp. 15–41.

[55] Koutsoukou-Argyraki, A., Formalising Mathematics—in Praxis; A Mathematician's First Experiences with Isabelle/HOL and the Why and How of Getting Started. Jahresbericht der Deutschen Mathematiker-Vereinigung 123, 3–26, 2021.

[56] Koutsoukou-Argyraki, A. & Li, W., Irrational rapidly convergent series, Archive of Formal Proofs, 2018, 23 May 2018.

[57] Koutsoukou-Argyraki, A. & Li, W., The transcendence of certain infinite series, Archive of Formal Proofs, 2019, 27 May 2019.

[58] Koutsoukou-Argyraki, A. & Li, W., Irrationality Criteria for Series by Erdős and Straus, Archive of Formal Proofs, 2020, 12 May 2020.

[59] Koutsoukou-Argyraki, A., Li, W., & Paulson, L. C., Irrationality and Transcendence Criteria for Infinite Series in Isabelle/HOL. To appear in: Experimental Mathematics (doi: 10.1080/10586458.2021.1980465).

[60] Larson, J. A., A short proof of a partition theorem for the ordinal ω^ω. Annals of Mathematical Logic, 6, 129–145, 1973.

[61] Li, W., Yu, L., Wu, Y., & Paulson, L. C., IsarStep: a Benchmark for High-level Mathematical Reasoning. In: *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021.* OpenReview.net, 2021.

[62] de Moura, L., Kong, S., Avigad, J., van Doorn, F., & von Raumer, J., The Lean theorem prover (system description). In: Felty, A.P. & Middeldorp, A. (eds.), *Automated Deduction. CADE-25. 25th International Conference on Automated Deduction, Berlin, Germany, August 1–7, 2015, Proceedings.* Lecture Notes in Computer Science. Vol. 9195, Springer, 2015, pp. 378–388.

[63] Nash-Williams, C. St. J. A., On well-quasi-ordering transfinite sequences, Proceedings of the Cambridge Philosophical Society 61, 33–39, 1965.

[64] Nipkow, T., Paulson, L. C., & Wenzel, M., *Isabelle/HOL: A Proof Assistant for Higher-Order Logic.* Lecture Notes in Computer Science, Vol. 2283, Springer, 2002.

[65] Paulson, L. C., The Foundation of a Generic Theorem Prover. Journal of Automated Reasoning, 5:3, 363–397, 1989.

[66] Paulson, L. C., Formalising mathematics in simple type theory. In: Centrone, S., Kant, D., & Sarikaya, D. (eds.), *Reflections on the Foundations of Mathematics. Univalent Foundations, Set Theory and General Thoughts.* Synthese Library, Vol. 407, Springer, 2019, pp. 437–454.

[67] Paulson, L. C., Zermelo Fraenkel set theory in higher-order logic, Archive of Formal Proofs, 2019, 24 October 2019.

[68] Paulson, L. C., The Nash-Williams partition theorem, Archive of Formal Proofs, 2020, 16 May 2020.

[69] Paulson, L. C., Ordinal partitions, Archive of Formal Proofs, 2020, 3 August 2020.

[70] Paulson, L. C., Wetzel's Problem and the Continuum Hypothesis, Archive of Formal Proofs, 2022, 18 February 2022.

[71] Phillips, N. C., & Weaver, N., The Calkin algebra has outer automorphisms. Duke Mathematical Journal, 139:1, 185–202, 2007.

[72] Polu, S. & Sutskever, I., Generative Language Modeling for Automated Theorem Proving. Preprint, 2020 (arXiv:2009.03393v1).

[73] Scholze, P., Perfectoid spaces. Publications mathématiques de l'Institut des Hautes Études Scientifiques 116, 245–313, 2012.

[74] Scholze, P., Liquid tensor experiment. Blog post on the Xena project blog, 5 December 2020.

[75] Scholze, P., Half a year of the Liquid Tensor Experiment: Amazing developments. Blog post on the Xena project blog, 5 June 2021.

[76] Shelah, S., Infinite abelian groups, Whitehead problem and some constructions. Israel Journal of Mathematics, 18, 243–256, 1974.

[77] Stathopoulos, Y., Koutsoukou-Argyraki, A., & Paulson, L. C., SErAPIS: A Concept-Oriented Search Engine for the Isabelle Libraries Based on Natural Language. Paper published on the website of the Isabelle Workshop 2020, 2020.

[78] Stathopoulos, Y., Koutsoukou-Argyraki, A. & Paulson, L. C., Developing a Concept-Oriented Search Engine for Isabelle Based on Natural Language: Technical Challenges. Paper published on the website of the 5th Conference on Artificial Intelligence and Theorem Proving, AITP 2020, 2020.

[79] Verchinine, K., Lyaletski, A., & Paskevich, A., System for Automated Deduction (SAD): A Tool for Proof Verification. In: Pfenning, F. (ed.) *Automated Deduction. CADE-21. 21st International Conference on Automated Deduction, Bremen, Germany, July 17–20, 2007, Proceedings.* Lecture Notes in Computer Science, Vol. 4603, Springer, 2007, pp. 398–403.

[80] Whitehead, A. N. & Russell, B., *Principia Mathematica, Vol. 1, 2, 3.* Cambridge University Press, 1927.

[81] Wiedijk, F. (ed.), *The Seventeen Provers of the World.* Lecture Notes in Artificial Intelligence, Vol. 3600, Springer, 2006.